M 11

```
DDDDDDDDDDDD        CCCCCCCCCCCC  LLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
```

L 3

```
**FILE**ID**KEYPAD




KK      KK  EEEEEEEEEE  YY      YY  PPPPPPPP      AAAAAA    DDDDDDDD
KK      KK  EEEEEEEEEE  YY      YY  PPPPPPPP      AAAAAA    DDDDDDDD
KK      KK  EE          YY      YY  PP      PP  AA      AA  DD      DD
KK      KK  EE          YY      YY  PP      PP  AA      AA  DD      DD
KK    KK    EE            YY  YY    PP      PP  AA      AA  DD      DD
KK    KK    EE            YY  YY    PP      PP  AA      AA  DD      DD
KKKKKK      EEEEEEEE        YY      PPPPPPPP    AA      AA  DD      DD
KKKKKK      EEEEEEEE        YY      PPPPPPPP    AA      AA  DD      DD
KK    KK    EE              YY      PP          AAAAAAAAAA  DD      DD
KK    KK    EE              YY      PP          AAAAAAAAAA  DD      DD
KK      KK  EE              YY      PP          AA      AA  DD      DD      ....
KK      KK  EE              YY      PP          AA      AA  DD      DD      ....
KK      KK  EEEEEEEEEE      YY      PP          AA      AA  DDDDDDDD        ....
KK      KK  EEEEEEEEEE      YY      PP          AA      AA  DDDDDDDD        ....


LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
Table of contents
```

```
0000            .TITLE  KEYPAD - KEYPAD SYMBOL TABLE MANIPULATION ROUTINES
0000            .IDENT  'V04-000'
0000
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000
0000    ; KEYPAD SYMBOL TABLE MANIPULATION ROUTINES
0000    ;
0000    ; AUTHOR: Peter George 15-March-1983
0000    ;
0000    ;       These routines all assume the structure of a keypad symbol
0000    ;       block, starting with the symbol name (SYM_T_SYMBOL), is as follows:
0000    ;
0000    ;               Byte - length of symbol name
0000    ;               ASCII symbol name
0000    ;               Word - combined lengths of next three strings + 4
0000    ;               Byte - length of if_state string
0000    ;               ASCII if_state string
0000    ;               Word - length of symbol value
0000    ;               ASCII symbol value
0000    ;               Byte - length of set_state string
0000    ;               ASCII set_state string
0000    ;
0000    ; MODIFIED BY:
0000    ;
0000    ;       V03-006 HWS0058         Harold Schultz          18-Apr-1984
0000    ;               Use DCL$ALLOC_STATE when setting temporary key states
0000    ;               and DCL$LOCKED_STATE to restore original state.
0000    ;               Exchange the synonym keypad names with the common key names.
0000    ;               (i.e. translate "FIND" to "E1" instead of "E1" to "FIND")
0000    ;
0000    ;       V03-006 HWS0052         Harold Schultz          09-Apr-1984
0000    ;               Translate synonym keypad names to a common key name.
0000    ;               (i.e. Translate "E1" to "FIND")
0000    ;               Add SHOW KEY/LOG.
0000    ;
0000    ;       V03-005 PCG0005         Peter George            09-Feb-1984
```

```
0000        ;                     Change format of SHOW KEY display.
0000        ;                     Add SHOW KEY/FULL.
0000        ;                     Zero the input buffer at the end of DCL$DEFKEY.
0000        ;
0000        ;             V03-004 PCG0004         Peter George            01-Dec-1983
0000        ;                     Add /ERASE.
0000        ;
0000        ;             V03-003 PCG0003         Peter George            27-Jul-1983
0000        ;                     Move PSECT declaration.
0000        ;
0000        ;             V03-002 PCG0002         Peter George            27-May-1983
0000        ;                     Validate key names before defining them.
0000        ;
0000        ;             V03-001 PCG0001         Peter George            07-Apr-1983
0000        ;                     Tolerate omission of SET KEY qualifiers.
0000        ;                     Add SHOW KEY/BRIEF/DIRECTORY.
0000        ;---
```

```
                                     0000   ;
                                     0000   ; MACRO LIBRARY CALLS
                                     0000   ;
                                     0000           PRCDEF                          ;DEFINE PROCESS DATA STRUCTURE
                                     0000           WRKDEF                          ;DEFINE COMMAND DATA STRUCTURE
                                     0000           PTRDEF                          ;DEFINE TOKEN DESCRIPTORS
                                     0000           SYMDEF                          ;DEFINE SYMBOL ENTRY OFFSETS
                                     0000           $CLIMSGDEF                      ;DEFINE ERROR/STATUS VALUES
                                     0000           $STSDEF                         ;DEFINE STATUS LONGWORD
                                     0000
                                 00000000           .PSECT  DCL$ZCODE,BYTE,RD,NOWRT
                                     0000
                                     0000   ;
                                     0000   ; ASCIC TEXT STRINGS FOR SHOW KEYS DISPLAY.
                                     0000   ;
                                     0000   SHOWHDR:
20 64 61 70 79 65 6B 20 43 41 21 00' 0000           .ASCIC  '!AC keypad definitions:'
3A 73 6E 6F 69 74 69 6E 69 66 65 64  000C
                                  17 0000
                                     0018   BRIEFFAO:
41 21 22 20 3D 20 53 41 21 20 20 00' 0018           .ASCIC  '  !AS = "!AS"'
                               22 53 0024
                                  0D 0018
                                     0026   FULLFAO:
41 21 22 20 3D 20 53 41 21 20 20 00' 0026           .ASCIC  '  !AS = "!AS"  (!ACecho,!ACterminate,!ACerase,!AClock!AC!AC!AS)'
6F 68 63 65 43 41 21 28 20 20 22 53  0032
74 61 6E 69 6D 72 65 74 43 41 21 2C  003E
21 2C 65 73 61 72 65 43 41 21 2C 65  004A
43 41 21 43 41 21 6B 63 6F 6C 43 41  0056
                         29 53 41 21 0062
                                  3F 0026
                            6F 6E 00' 0066  NO:     .ASCIC  'no'
                                  02 0066
                3D 65 74 61 74 73 00' 0069  STATE:  .ASCIC  'state='
                                  06 0069
                                  00 0070   NULL:   .BYTE   0
                               2C 00' 0071  COMMA:  .ASCIC  ','
                                  01 0071
                                     0073   ;
                                     0073   ; SYNONYM KEY NAME TABLES
                                     0073   ;
                                     0073   ;
                                     0073   ; DEFINE SYNONYM KEY NAMES
                                     0073   ;
                                     0073   ; SYNONYM_NAME  SETS UP THE RELATIONSHIP BETWEEN THE SYNONYM (NAME1) AND
                                     0073   ; THE COMMON KEY NAME (NAME2) THAT THE SYNONYM IS TRANSLATED TO. IF A NEW
                                     0073   ; SYNONYM IS CREATED THAT TRANSLATES TO AN EXISTING COMMON KEY NAME (IN
                                     0073   ; SYNDEF_TAB), ONLY AN ENTRY IN SYNNAME_TAB NEEDS TO BE ADDED. IF A NEW
                                     0073   ; COMMON KEY NAME IS NEEDED, THEN ADD IT TO SYNDEF_TAB.
                                     0073   ;
                                     0073           .MACRO SYNONYM_NAME NAME1,NAME2
                                     0073           .ASCIC  "NAME1"
                                     0073           .WORD   'NAME2'_ADR - SYNDEF_TAB
                                     0073           .ENDM
                                     0073   ;
                                     0073   ;
                                     0073   SYNNAME_TAB:
```

```
      0073          SYNONYM_NAME    FIND,E1             ;DEFINE SYNONYM KEY NAMES
      007A          SYNONYM_NAME    INSERT_HERE,E2
      0088          SYNONYM_NAME    REMOVE,E3
      0091          SYNONYM_NAME    SELECT,E4
      009A          SYNONYM_NAME    PREV_SCREEN,E5
      00A8          SYNONYM_NAME    NEXT_SCREEN,E6
  00  00B6          .BYTE           0                   ;END OF TABLE MARKER
      00B7  ;
      00B7  ; DEFINE COMMON SYNONYM KEY NAMES
      00B7  ;
      00B7  ; NAME = COMMON TRANSLATED NAME (I.E. "FIND")
      00B7  ;
      00B7          .MACRO  SYNONYM_TRN     NAME
      00B7  'NAME'_ADR:
      00B7          .ASCIC  "NAME"
      00B7          .ENDM
      00B7
      00B7
      00B7  SYNDEF_TAB:
      00B7          SYNONYM_TRN     E1
      00BA          SYNONYM_TRN     E2
      00BD          SYNONYM_TRN     E3
      00C0          SYNONYM_TRN     E4
      00C3          SYNONYM_TRN     E5
      00C6          SYNONYM_TRN     E6
      00C9
```

```
                         00C9          .SBTTL  DEFINE KEYPAD SYMBOL
                         00C9   ;+
                         00C9   ; DCL$DEFKEY - DEFINE KEYPAD SYMBOL
                         00C9   ;
                         00C9   ; THIS ROUTINE IS CALLED AS AN INTERNAL COMMAND TO EXECUTE THE DEFINE/KEY
                         00C9   ; COMMAND.
                         00C9   ;
                         00C9   ; INPUTS:
                         00C9   ;
                         00C9   ;       R8 = ADDRESS OF SCRATCH BUFFER DESCRIPTOR.
                         00C9   ;       R9 = ADDRESS OF SCRATCH STACK.
                         00C9   ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                         00C9   ;       R11 = BASE ADDRESS OF PROCESS WORK AREA.
                         00C9   ;
                         00C9   ; OUTPUTS:
                         00C9   ;
                         00C9   ;       THE SPECIFIED META-KEY IS ASSIGNED TO THE SPECIFIED EQUIVALENCE
                         00C9   ;       STRING.
                         00C9   ;-
                         00C9
                         00C9   DCL$DEFKEY::                                    ;DEFINE META-KEY EQUIVALENCE
                         00C9
                         00C9   ;
                         00C9   ; SET INITIAL PARSE STATE.
                         00C9   ;
         56     01  D0   00C9           MOVL    #SYM_M_ECHO,R6                  ;INITIALIZE KEYPAD FLAGS
         57     01  D0   00CC           MOVL    #1,R7                           ;INITIALIZE LOCAL FLAGS (/LOG)
                58  D4   00CF           CLRL    R8                              ;CLEAR IF_STATE TOKEN PTR
                79  7C   00D1           CLRQ    -(R9)                           ;ALLOCATE SET_STATE DESCRIPTOR
                         00D3
                         00D3   ;
                         00D3   ; PROCESS THE TOKENS ON THE COMMAND LINE.
                         00D3   ;
             FF2A'  30   00D3           BSBW    DCL$GETDVAL                     ;SKIP PAST /KEY DESCRIPTOR
             FF27'  30   00D6   10$:    BSBW    DCL$GETDVAL                     ;GET NEXT DESCRIPTOR VALUES
         55     04  91   00D9           CMPB    #PTR_K_ENDLINE,R5               ;EOL?
                03  12   00DC           BNEQ    15$                             ;NO, CONTINUE PARSING
              00AF  31   00DE           BRW     70$                             ;YES, THEN DONE PARSING
         55     03  91   00E1   15$:    CMPB    #PTR_K_PARAMETR,R5              ;ITEM TYPE PARAMETER?
                05  12   00E4           BNEQ    20$                             ;NO, THEN PROCESS QUALIFIER
         79     51  7D   00E6           MOVQ    R1,-(R9)                        ;SAVE PARAMETER DESCRIPTOR
                EB  11   00E9           BRB     10$                             ;GET NEXT TOKEN
             FF12'  30   00EB   20$:    BSBW    DCL$GETNVAL                     ;GET QUALIFIER NUMBER
00000000'8F     51  D1   00EE           CMPL    R1,#CLI$K_DEFK_TERM             ;QUALIFIER MATCH?
                38  13   00F5           BEQL    30$                             ;YES, THEN PROCESS
00000000'8F     51  D1   00F7           CMPL    R1,#CLI$K_DEFK_ECHO             ;QUALIFIER MATCH?
                3B  13   00FE           BEQL    35$                             ;YES, THEN PROCESS
00000000'8F     51  D1   0100           CMPL    R1,#CLI$K_DEFK_LOCK             ;QUALIFIER MATCH?
                3E  13   0107           BEQL    40$                             ;YES, THEN PROCESS
00000000'8F     51  D1   0109           CMPL    R1,#CLI$K_DEFK_LOG              ;QUALIFIER MATCH?
                41  13   0110           BEQL    43$                             ;YES, THEN PROCESS
00000000'8F     51  D1   0112           CMPL    R1,#CLI$K_DEFK_SET_             ;QUALIFIER MATCH?
                4F  13   0119           BEQL    45$                             ;YES, THEN PROCESS
00000000'8F     51  D1   011B           CMPL    R1,#CLI$K_DEFK_IF_S             ;QUALIFIER MATCH?
                58  13   0122           BEQL    50$                             ;YES, THEN PROCESS
00000000'8F     51  D1   0124           CMPL    R1,#CLI$K_DEFK_ERAS             ;QUALIFIER MATCH?
                31  13   012B           BEQL    55$                             ;YES, THEN PROCESS
```

```
                    A7   11  012D   204 25$:    BRB     10$                                 ;GET NEXT
                             012F   205
                             012F   206 30$:    SETBIT  SYM_V_TERMINATE,R6                  ;ASSUME /TERMINATE
              F7 53   00  E1  0132   207         BBC     #PTR_V_NEGATE-PTR_V_FLAGS,R3,25$;IGNORE IF NOT /NOTERMINATE
                             0136   208         CLRBIT  SYM_V_TERMINATE,R6                  ;CLEAR TERMINATE FLAG
                    F2   11  0139   209         BRB     25$                                 ;GET NEXT
                             013B   210 35$:    SETBIT  SYM_V_ECHO,R6                       ;ASSUME /ECHO
              EB 53   00  E1  013E   211         BBC     #PTR_V_NEGATE-PTR_V_FLAGS,R3,25$;IGNORE IF NOT /NOECHO
                             0142   212         CLRBIT  SYM_V_ECHO,R6                       ;CLEAR ECHO FLAG
                    E6   11  0145   213         BRB     25$                                 ;GET NEXT
                             0147   214 40$:    SETBIT  SYM_V_LOCK,R6                       ;ASSUME /LOCK
              DF 53   00  E1  014A   215         BBC     #PTR_V_NEGATE-PTR_V_FLAGS,R3,25$;IGNORE IF NOT /NOLOCK
                             014E   216         CLRBIT  SYM_V_LOCK,R6                       ;CLEAR LOCK FLAG
                    DA   11  0151   217         BRB     25$                                 ;GET NEXT
                 57   01  D0  0153   218 43$:    MOVL    #1,R7                               ;ASSUME /LOG
              D3 53   00  E1  0156   219         BBC     #PTR_V_NEGATE-PTR_V_FLAGS,R3,25$;IGNORE IF NOT /NOLOG
                      57  D4  015A   220         CLRL    R7                                  ;CLEAR FLAG
                      CF  11  015C   221         BRB     25$                                 ;GET NEXT
                             015E   222 55$:    SETBIT  SYM_V_ERASE,R6                      ;ASSUME /ERASE
              C8 53   00  E1  0161   223         BBC     #PTR_V_NEGATE-PTR_V_FLAGS,R3,25$;IGNORE IF NOT /NOERASE
                             0165   224         CLRBIT  SYM_V_ERASE,R6                      ;CLEAR ERASE FLAG
                    C3   11  0168   225         BRB     25$                                 ;GET NEXT
                             016A   226
                             016A   227 45$:    CLRBIT  SYM_V_STATE,R6                      ;ASSUME /NOSET_STATE
              BC 53   00  E0  016D   228         BBS     #PTR_V_NEGATE-PTR_V_FLAGS,R3,25$;IGNORE IF NOT /SET_STATE
                             0171   229         SETBIT  SYM_V_STATE,R6                      ;SET STATE FLAG
                  FE89'  30  0174   230         BSBW    DCL$GETDVAL                         ;GET THE ASSOCIATED VALUE
                 69   51  7D  0177   231         MOVQ    R1,(R9)                             ;SAVE THAT VALUE
                      B1  11  017A   232         BRB     25$                                 ;GET NEXT
                             017C   233
                      58  D4  017C   234 50$:    CLRL    R8                                  ;ASSUME /NOIF_STATE
              AB 53   00  E0  017E   235         BBS     #PTR_V_NEGATE-PTR_V_FLAGS,R3,25$;IGNORE IF NOT /IF_STATE
              58   BA AA  D0  0182   236         MOVL    WRK_L_RSLNXT(R10),R8                ;SAVE VALUE TOKEN PTR
                  FE77'  30  0186   237 52$:    BSBW    DCL$GETDVAL                         ;GET NEXT DESCRIPTOR VALUE
                 54   05  D1  0189   238         CMPL    #PTR_K_COMMA,R4                     ;TERMINATOR A COMMA?
                      F8  13  018C   239         BEQL    52$                                 ;GET NEXT VALUE
                      9D  11  018E   240         BRB     25$                                 ;GET NEXT
                             0190   241
                             0190   242 ;
                             0190   243 ; INSERT THE META-KEY SYMBOL IN THE SPECIFIED KEYPAD SYMBOL TABLES.
                             0190   244 ;
                             0190   245 ; SCRATCH STACK LOOKS LIKE:
                             0190   246 ;       (R9)    EQUIVALENCE STRING DESCRIPTOR
                             0190   247 ;       8(R9)   META-KEY NAME DESCRIPTOR
                             0190   248 ;       16(R9)  SET_STATE DESCRIPTOR
                             0190   249 ; R6 CONTAINS SYMBOL FLAGS
                             0190   250 ;
                 08 A9   7F  0190   251 70$:    PUSHAQ  8(R9)                               ;PUSH THE DESCRIPTOR ADDRESS
         00000000'EF   01  FB  0193   252         CALLS   #1,VALIDATE_KEY_NAME                ;IS IT VALID?
                 5F   50  E9  019A   253         BLBC    R0,97$                              ;NO, THEN RETURN ERROR
              50      57  D0  019D   254         MOVL    R7,R0                               ;GET /LOG FLAG
           51      08 A9  7D  01A0   255         MOVQ    8(R9),R1                            ;GET KEY NAME DESCRIPTOR
                  04AA   30  01A4   256         BSBW    DCL$SYNONYM                         ;CHECK FOR SYNONYMS
              08 A9   51  7D  01A7   257         MOVQ    R1,8(R9)                            ;SAVE RETURNED KEYPAD NAME
              03 56   01  E0  01AB   258         BBS     #SYM_V_TERMINATE,R6,71$             ;BRANCH IF /TERMINATE
                 56   01  CA  01AF   259         BICL    #SYM_M_ECHO,R6                      ;IGNORE THE ECHO FLAG
                             01B2   260 71$:    ASSUME  PTR_K_COMMA NE 0
```

```
        BA AA    58  D0  01B2           MOVL    R8,WRK_L_RSLNXT(R10)           ;RESET FOR FIRST IF_STATE VALUE
                 09  13  01B6           BEQL    75$                            ;SKIP IF NONE
              FE45'  30  01B8  72$:     BSBW    DCL$GETDVAL                    ;GET NEXT DESCRIPTOR VALUE
           58    54  D0  01BB           MOVL    R4,R8                          ;SAVE THE TERMINATOR
               05D3  30  01BE           BSBW    DCL$ALLOC_STATE                ;SET NEW STATE
               03F9  30  01C1  75$:     BSBW    DCL$ALLOCKEY                   ;ALLOCATE THE KEYPAD SYMBOL
              31 50  E9  01C4           BLBC    R0,95$                         ;BRANCH IF ERROR
                         01C7
                         01C7  ;
                         01C7  ; OUTPUT /LOG MESSAGE.
                         01C7  ;
              13 57  E9  01C7           BLBC    R7,80$                         ;SKIP IF /NOLOG
              08 A9  9F  01CA           PUSHAB  8(R9)                          ;SET ADDRESS OF META-KEY NAME DESCR
              48 AB  DD  01CD           PUSHL   PRC_L_CURRKEY(R11)             ;SET ADDRESS OF ASCIC STATE NAME
           51    02  D0  01D0           MOVL    #2,R1                          ;SET ARGUMENT COUNT
     50  0003DDC3 8F  D0  01D3          MOVL    #CLI$_DEFKEY,R0                ;SET STATUS
              FE23'  30  01DA           BSBW    DCL$FORMMSG                    ;OUTPUT THE LOG MESSAGE
           58    05  D1  01DD  80$:     CMPL    #PTR_K_COMMA,R8                ;TERMINATOR A COMMA?
                 05  12  01E0           BNEQ    90$                            ;NO, TIME TO EXIT
              FE1B'  30  01E2           BSBW    DCL$LOCKED_STATE               ;YES, RESTORE LOCKED KEY STATE
                         01E5                                                  ;  BEFORE GETTING NEXT STATE
                 D1  11  01E5           BRB     72$                            ;GET NEXT VALUE
                         01E7
                         01E7  ;
                         01E7  ; RESTORE KEYPAD STATE AND RETURN.
                         01E7  ;
 0100 8F   00   6E   00  2C  01E7  90$: MOVC5   #0,(SP),#0,#WRK_C_INPBUFSIZ,-  ;RESET THE INPUT BUFFER
            F896 CA      01EE                   WRK_G_INPBUF(R10)              ;
                         01F1           STATUS  NORMAL                         ;SET NORMAL COMPLETION
              FE05'  30  01F8  95$:     BSBW    DCL$LOCKED_STATE               ;RESTORE KEY STATE
                     05  01FB           RSB                                    ;
     50  00038280 8F  D0  01FC  97$:    MOVL    #CLI$_IVKEYNAM,R0              ;SET STATUS
                 F3  11  0203           BRB     95$                            ;RETURN
```

```
                        0205            .SBTTL  DELETE KEYPAD SYMBOL
                        0205    ;+
                        0205    ; DCL$DELKEY - DELETE KEYPAD SYMBOL
                        0205    ;
                        0205    ; THIS ROUTINE IS CALLED AS AN INTERNAL COMMAND TO EXECUTE THE DELETE/KEY
                        0205    ; COMMAND.
                        0205    ;
                        0205    ; INPUTS:
                        0205    ;
                        0205    ;       R8 = ADDRESS OF SCRATCH BUFFER DESCRIPTOR.
                        0205    ;       R9 = ADDRESS OF SCRATCH STACK.
                        0205    ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                        0205    ;       R11 = BASE ADDRESS OF PROCESS WORK AREA.
                        0205    ;
                        0205    ; OUTPUTS:
                        0205    ;
                        0205    ;       THE SPECIFIED META-KEY IS DELETED FROM THE SYMBOL TABLE.
                        0205    ;-
                        0205
                        0205    DCL$DELKEY::                                            ;DELETE KEYPAD DEFINITION
                        0205    ;
                        0205    ; SET INITIAL PARSE STATE.
                        0205    ;
              58    D4  0205            CLRL    R8                                      ;CLEAR STATE TOKEN PTR
              03    DD  0207            PUSHL   #3                                      ;INITIALIZE LOCAL FLAGS
                        0209                                                            ; /LOG, /ALL, NO UNDEFINED SYMBOLS
                        0209
                        0209    ;
                        0209    ; PROCESS THE TOKENS ON THE COMMAND LINE.
                        0209    ;
           FDF4'    30  0209            BSBW    DCL$GETDVAL                             ;SKIP PAST /KEY DESCRIPTOR
           FDF1'    30  020C    10$:    BSBW    DCL$GETDVAL                             ;GET NEXT DESCRIPTOR VALUES
          55  04    91  020F            CMPB    #PTR_K_ENDLINE,R5                       ;EOL?
              03    12  0212            BNEQ    15$                                     ;NO, CONTINUE PARSING
            0044    31  0214            BRW     50$                                     ;YES, THEN DONE PARSING
          55  03    91  0217    15$:    CMPB    #PTR_K_PARAMETR,R5                      ;ITEM TYPE PARAMETER?
              08    12  021A            BNEQ    20$                                     ;NO, THEN PROCESS QUALIFIER
          56  51    7D  021C            MOVQ    R1,R6                                   ;SAVE PARAMETER DESCRIPTOR
          6E  02    CA  021F            BICL    #2,(SP)                                 ;CLEAR /ALL FLAG
              E8    11  0222            BRB     10$                                     ;GET NEXT TOKEN
           FDD9'    30  0224    20$:    BSBW    DCL$GETNVAL                             ;GET QUALIFIER NUMBER
 00000000'8F  51    D1  0227            CMPL    R1,#CLI$K_DELK_LOG                      ;QUALIFIER MATCH?
              0B    13  022E            BEQL    25$                                     ;YES, THEN PROCESS
 00000000'8F  51    D1  0230            CMPL    R1,#CLI$K_DELK_STAT                     ;QUALIFIER MATCH?
              0E    13  0237            BEQL    30$                                     ;YES, THEN PROCESS
              D1    11  0239            BRB     10$                                     ;GET NEXT
                        023B
          6E  01    C8  023B    25$:    BISL    #1,(SP)                                 ;ASSUME /LOG
       CA 53  00    E1  023E            BBC     #PTR_V_NEGATE-PTR_V_FLAGS,R3,10$;IGNORE IF NOT /NOLOG
          6E  01    CA  0242            BICL    #1,(SP)                                 ;CLEAR FLAG
              C5    11  0245            BRB     10$                                     ;GET NEXT
                        0247
              58    D4  0247    30$:    CLRL    R8                                      ;ASSUME /NOSTATE
       BF 53  00    E0  0249            BBS     #PTR_V_NEGATE-PTR_V_FLAGS,R3,10$;IGNORE IF NOT /STATE
     58    BA AA    D0  024D            MOVL    WRK_L_RSLNXT(R10),R8                    ;SAVE VALUE TOKEN PTR
           FDAC'    30  0251    32$:    BSBW    DCL$GETDVAL                             ;GET NEXT DESCRIPTOR VALUE
          54  05    D1  0254            CMPL    #PTR_K_COMMA,R4                         ;TERMINATOR A COMMA?
```

```
                F8   13  0257   352          BEQL    32$                              ;GET NEXT VALUE
                B1   11  0259   353          BRB     10$                              ;GET NEXT
                         025B   354
                         025B   355 ;
                         025B   356 ; SET KEYPAD STATE.
                         025B   357 ;
                         025B   358 50$:     ASSUME  PTR_K_COMMA NE 0
         BA AA  58   D0  025B   359          MOVL    R8,@RK_L_RSLNXT(R10)             ;RESET FOR FIRST STATE VALUE
                09   13  025F   360          BEQL    53$                              ;SKIP IF NONE
             FD9C'   30  0261   361 52$:     BSBW    DCL$GETDVAL                      ;GET NEXT DESCRIPTOR VALUE
            58  54   D0  0264   362          MOVL    R4,R8                            ;SAVE THE TERMINATOR
              052A   30  0267   363          BSBW    DCL$ALLOC_STATE                  ;SET NEW STATE
                         026A   364 ;
                         026A   365 ; DETERMINE WHETHER DELETING ONE SYMBOL OR /ALL SYMBOLS.
                         026A   366 ;
         40 6E  01   E0  026A   367 53$:     BBS     #1,(SP),60$                      ;BRANCH IF /ALL
                         026E   368
                         026E   369 ;
                         026E   370 ; FIND SPECIFIED SINGLE SYMBOL VALUE.
                         026E   371 ;
            51  56   7D  026E   372 54$:     MOVQ    R6,R1                    ;GET DESCRIPTOR OF SYMBOL NAME
            50  6E   D0  0271   373          MOVL    (SP),R0                  ;GET /LOG FLAG
              03DA   30  0274   374          BSBW    DCL$SYNONYM              ;CHECK FOR SYNONYM KEY NAME
            56  51   7D  0277   375          MOVQ    R1,R6                    ;SAVE DESCRIPTOR IN CASE OF UNDKEY
              0461   30  027A   376          BSBW    DCL$FIND_KEYPAD          ;SEARCH FOR SYMBOL
             08 50   E8  027D   377          BLBS    R0,55$                   ;BRANCH IF SUCCESSFUL
                         0280   378
                         0280   379 ;
                         0280   380 ; OUTPUT WARNING MESSAGE.
                         0280   381 ;
            6E  04   C8  0280   382          BISL    #4,(SP)                  ;SET UNDEFINED SYMBOL FLAG
              0078   30  0283   383          BSBW    UNDKEY                   ;OUTPUT UNDEFINED KEY MSG
                03   11  0286   384          BRB     58$                      ;GET NEXT TABLE
                         0288   385
                         0288   386 ;
                         0288   387 ; OUTPUT LOG MESSAGE IF REQUESTED.  DELETE THE SYMBOL.
                         0288   388 ;
              0057   30  0288   389 55$:     BSBW    DELKEY                   ;OUTPUT DELKEY MSG AND DELETE THE KEY
            58  05   D1  028B   390 58$:     CMPL    #PTR_K_COMMA,R8          ;TERMINATOR A COMMA?
                05   12  028E   391          BNEQ    90$                      ;NO, TIME TO EXIT
             FD6D'   30  0290   392          BSBW    DCL$LOCKED_STATE         ;YES, RESTORE LOCKED KEY STATE
                         0293   393                                           ;  BEFORE GETTING NEXT STATE
                CC   11  0293   394          BRB     52$                      ;GET NEXT STATE
                         0295   395
                         0295   396 ;
                         0295   397 ; RESTORE KEYPAD STATE, SET STATUS, AND EXIT.
                         0295   398 ;
                         0295   399 90$:     STATUS  NORMAL                   ;ASSUME SUCCESSFUL COMPLETION
            51  8E   D0  029C   400          MOVL    (SP)+,R1                 ;GET FLAGS
         07 51  02   E1  029F   401          BBC     #2,R1,95$                ;BRANCH IF NO UNDEFINED SYMBOLS
50   10038260 8F     D0  02A3   402          MOVL    #CLI$_UNDKEY!STS$M_INHIB_MSG,R0 ;SET STATUS, INHIBIT RESIGNAL
             FD53'   30  02AA   403 95$:     BSBW    DCL$LOCKED_STATE         ;RESTORE KEY STATE
                     05  02AD   404          RSB
                         02AE   405
                         02AE   406 ;
                         02AE   407 ; DELETE ALL SYMBOL ENTRIES FOR THE SPECIFIED OR CURRENT STATE.
                         02AE   408 ;
```

```
  56   40 AB    7E  02AE    60$:    MOVAQ   PRC_Q_KEYPAD(R11),R6    ;GET ADDRESS OF KEYPAD SYMBOL TABLE
     5C    56    D0  02B2            MOVL    R6,AP                   ;COPY ADDRESS OF TABLE LISTHEAD
                     02B5
                     02B5    ;
                     02B5    ; GET NEXT SYMBOL.
                     02B5    ;
     56    66    D0  02B5    70$:    MOVL    (R6),R6                 ;GET ADDRESS OF NEXT ENTRY
     5C    56    D1  02B8            CMPL    R6,AP                   ;END OF TABLE?
           CE    13  02BB            BEQL    58$                     ;IF EQL YES
                     02BD
                     02BD    ;
                     02BD    ; IF STATE DOES NOT MATCH, THEN SKIP THIS SYMBOL.
                     02BD    ;
  54   0C A6    9E  02BD            MOVAB   SYM_T_SYMBOL(R6),R4     ;GET ADDRESS OF SYMBOL NAME
     51    84    9A  02C1            MOVZBL  (R4)+,R1                ;GET LENGTH OF SYMBOL NAME
 54  02 A441    9E  02C4            MOVAB   2(R4)[R1],R4            ;GET ADDRESS OF IF_STATE
  52   48 AB    D0  02C9            MOVL    PRC_L_CURRKEY(R11),R2   ;GET CURRENT STATE LENGTH/ADDRESS
     51    82    9A  02CD            MOVZBL  (R2)+,R1                ;
     84    51    91  02D0            CMPB    R1,(R4)+                ;STATE LENGTH THE SAME?
           E0    12  02D3            BNEQ    70$                     ;IF DIFF THEN GET NEXT
 64  62    51    29  02D5            CMPC    R1,(R2),(R4)            ;STATES MATCH?
           DA    12  02D9            BNEQ    70$                     ;NO, THEN GET NEXT
                     02DB
                     02DB    ;
                     02DB    ; STATE DID MATCH.  OUTPUT LOG MESSAGE IF REQUESTED.  DELETE THE SYMBOL.
                     02DB    ;
     53    56    D0  02DB            MOVL    R6,R3                   ;COPY SYMBOL ADDRESS
           02    10  02DE            BSBB    DELKEY                  ;OUTPUT LOG MESSAGE AND DELETE THE KEY
           CC    11  02E0            BRB     60$                     ;GET NEXT
```

```
                                02E2     439 ;+
                                02E2     440 ; DELKEY - OUTPUT THE DELKEY /LOG MESSAGE AND DELETE THE SPECIFIED KEY.
                                02E2     441 ;-
                                02E2     442 DELKEY:                                             ;OUTPUT DELKEY MSG AND DELETE THE KEY
             14 04 AE     E9    02E2     443             BLBC    4(SP),80$                       ;SKIP IF /NOLOG SPECIFIED
          7E    0C A3     9E    02E6     444             MOVAB   SYM_T_SYMBOL(R3),-(SP)          ;SET ADDR OF ASCIC STRING
                48 AB     DD    02EA     445             PUSHL   PRC_L_CURRKEY(R11)              ;SET ADDRESS OF ASCIC STATE NAME
             51    02     D0    02ED     446             MOVL    #2,R1                           ;SET FAO COUNT
    50    0003DDCB 8F     D0    02F0     447             MOVL    #CLI$_DELKEY,R0                 ;SET STATUS
                 FD06'    30    02F7     448             BSBW    DCL$FORMMSG                     ;OUTPUT THE MESSAGE
                 FD03'    30    02FA     449 80$:        BSBW    DCL$DEALLOCSYM                  ;DEALLOCATE KEYPAD ENTRY
                          05    02FD     450             RSB                                     ;RETURN
```

```
                         02FE     452 ;+
                         02FE     453 ; UNDKEY - OUTPUT THE UNDKEY WARNING MESSAGE.
                         02FE     454 ;-
                         02FE     455 UNDKEY:                                         ;OUTPUT UNDKEY WARNING
            7E    56  7D 02FE     456         MOVQ    R6,-(SP)                        ;PUSH DESC OF SYMBOL NAME
                  5E  DD 0301     457         PUSHL   SP                              ;PUSH DESCR ADDRESS
               48 AB  DD 0303     458         PUSHL   PRC_L_CURRKEY(R11)              ;PUSH ADDR OF ASCIC STATE
               51 02  D0 0306     459         MOVL    #2,R1                           ;SET FAO COUNT
   50   00038260 8F   D0 0309     460         MOVL    #CLI$_UNDKEY,R0                 ;SET UNDEFINED SYMBOL STATUS
               FCED'  30 0310     461         BSBW    DCL$FORMMSG                     ;OUTPUT THE MESSAGE
            5E    08  C0 0313     462         ADDL    #8,SP                           ;RESTORE THE STACK
                      05 0316     463         RSB                                     ;RETURN
```

```
                         0317          .SBTTL  SHOW KEYPAD SYMBOL TABLE ENTRIES
                         0317   ;+
                         0317   ; DCL$SHOWKEY - SHOW KEYPAD SYMBOL TABLE ENTRIES
                         0317   ;
                         0317   ; THIS ROUTINE IS CALLED AS AN INTERNAL COMMAND TO EXECUTE THE SHOW KEYS
                         0317   ; COMMAND.
                         0317   ;
                         0317   ; INPUTS:
                         0317   ;
                         0317   ;       R8 = ADDRESS OF SCRATCH BUFFER DESCRIPTOR.
                         0317   ;       R9 = ADDRESS OF SCRATCH STACK.
                         0317   ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                         0317   ;       R11 = BASE ADDRESS OF PROCESS WORK AREA.
                         0317   ;
                         0317   ; OUTPUTS:
                         0317   ;
                         0317   ;       THE SPECIFIED KEYPAD SYMBOL TABLE ENTRY OR ALL KEYPAD SYMBOL TABLE
                         0317   ;       ENTRIES FOR THE CURRENT OR SPECIFIED STATE ARE WRITTEN TO THE OUTPUT
                         0317   ;       STREAM.
                         0317   ;-
                         0317
                         0317   DCL$SHOWKEY::                                   ;SHOW KEYPAD SYMBOL TABLE ENTRIES
                         0317   ;
                         0317   ; INIT PARSE STATE.
                         0317   ; FLAG BITS ARE:    0 = /ALL, 1 = /BRIEF, 2 = FIRST STATE,
                         0317   ;                   3 = UNDKEY, 4 = /DIRECTORY.
                         0317   ;
             01     DD   0317           PUSHL   #1                      ;INIT MESSAGE FLAG (ASSUME /LOG)
             7E     D4   0319           CLRL    -(SP)                   ;CLEAR STATE TOKEN PTR
        7E   02     D0   031B           MOVL    #2,-(SP)                ;INIT FLAGS (ASSUME /ALL /BRIEF)
             56     7C   031E           CLRQ    R6                      ;ZERO DESCRIPTOR OF SYMBOL NAME
                         0320
                         0320   ;
                         0320   ; PROCESS THE TOKENS ON THE COMMAND LINE.
                         0320   ;
           FCDD'    30   0320   10$:    BSBW    DCL$GETDVAL             ;GET NEXT DESCRIPTOR VALUE
        55   04     91   0323           CMPB    #PTR_K_ENDLINE,R5       ;END OF LINE?
             11     13   0326           BEQL    11$                     ;BRANCH IF SO
        55   03     91   0328           CMPB    #PTR_K_PARAMETR,R5      ;PARAMETER?
             0F     12   032B           BNEQ    12$                     ;BRANCH IF QUALIFIER
                         032D           ASSUME  PTR_V_KEYWORD EQ 21
     EF 53   01     E0   032D           BBS     #1,R3,10$               ;SKIP IF OPTION KEYWORD
        56   51     7D   0331           MOVQ    R1,R6                   ;SAVE DESCRIPTOR OF SYMBOL NAME
        6E   01     C8   0334           BISL    #1,(SP)                 ;INDICATE NOT /ALL
             E7     11   0337           BRB     10$                     ;GET NEXT TOKEN
           007A     31   0339   11$:    BRW     20$                     ;AT END OF LINE. EXECUTE COMMAND
                         033C
                         033C   ;
                         033C   ; PROCESS QUALIFIERS.
                         033C   ;
           FCC1'    30   033C   12$:    BSBW    DCL$GETNVAL             ;GET QUALIFIER NUMBER
00000000'8F  51     D1   033F           CMPL    R1,#CLI$K_SHKY_FULL     ;/FULL QUALIFIER?
             48     13   0346           BEQL    16$                     ;YES, THEN PROCESS
00000000'8F  51     D1   0348           CMPL    R1,#CLI$K_SHKY_BRIE     ;/BRIEF QUALIFIER?
             4B     13   034F           BEQL    18$                     ;YES, THEN PROCESS
00000000'8F  51     D1   0351           CMPL    R1,#CLI$K_SHKY_DIRE     ;/DIRECTORY QUALIFIER?
             4F     13   0358           BEQL    19$                     ;YES, THEN PROCESS
```

```
00000000'8F    51   D1  035A              CMPL    R1,#CLISK_SHKY_LOG                 ;/LOG QUALIFIER?
               1F   13  0361              BEQL    14$                                ;YES, THEN PROCESS
00000000'8F    51   D1  0363              CMPL    R1,#CLISK_SHKY_STAT                ;/STATE QUALIFIER?
               B4   12  036A              BNEQ    10$                                ;NO, THEN IGNORE
                        036C
                        036C     ;
                        036C     ; PROCESS /STATE=state QUALIFIER.
                        036C     ;
          04 AE     D4  036C              CLRL    4(SP)                              ;ASSUME /NOSTATE
    AD 53    00     E0  036F              BBS     #PTR_V_NEGATE-PTR_V_FLAGS,R3,10$;IGNORE IF NOT /STATE
 04 AE    BA AA     D0  0373              MOVL    WRK_L_RSLNXT(R10),4(SP)            ;SAVE VALUE TOKEN PTR
           FC85'    30  0378     13$:     BSBW    DCL$GETDVAL                        ;GET NEXT DESCRIPTOR VALUE
       54    05     D1  037B              CMPL    #PTR_K_COMMA,R4                    ;TERMINATOR A COMMA?
             F8     13  037E              BEQL    13$                                ;GET NEXT VALUE
             9E     11  0380              BRB     10$                                ;GET NEXT
                        0382
                        0382     ;
                        0382     ; PROCESS /LOG QUALIFIER
                        0382     ;
    08 AE    01     C8  0382     14$:     BISL    #1,8(SP)                           ;ASSUME /LOG
    96 53    00     E1  0386              BBC     #PTR_V_NEGATE-PTR_V_FLAGS,R3,10$ ;IGNORE IF /NOLOG
    08 AE    01     CA  038A              BICL    #1,8(SP)                           ;SET FLAG TO /NOLOG
             90     11  038E              BRB     10$                                ;GET NEXT TOKEN
                        0390     ;
                        0390     ; PROCESS /FULL /BRIEF AND /DIRECTORY QUALIFIERS.
                        0390     ;
       6E    02     CA  0390     16$:     BICL    #2,(SP)                            ;ASSUME /FULL
    89 53    00     E1  0393              BBC     #PTR_V_NEGATE-PTR_V_FLAGS,R3,10$;IGNORE IF NOT /NOBRIEF
       6E    02     C8  0397              BISL    #2,(SP)                            ;SET FLAG
             84     11  039A              BRB     10$                                ;GET NEXT
       6E    02     C8  039C     18$:     BISL    #2,(SP)                            ;ASSUME /BRIEF
    10 53    00     E1  039F              BBC     #PTR_V_NEGATE-PTR_V_FLAGS,R3,190$;IGNORE IF NOT /NOBRIEF
       6E    02     CA  03A3              BICL    #2,(SP)                            ;CLEAR FLAG
          FF77      31  03A6              BRW     10$                                ;GET NEXT
       6E    10     C8  03A9     19$:     BISL    #16,(SP)                           ;ASSUME /DIRECTORY
    03 53    00     E1  03AC              BBC     #PTR_V_NEGATE-PTR_V_FLAGS,R3,190$;IGNORE IF NOT /NODIRECTORY
       6E    10     CA  03B0              BICL    #16,(SP)                           ;CLEAR FLAG
          FF6A      31  03B3     190$:    BRW     10$                                ;GET NEXT
                        03B6
                        03B6     ;
                        03B6     ; EXECUTE /DIRECTORY.
                        03B6     ;
 3E 6E       04     E1  03B6     20$:     BBC     #4,(SP),21$                        ;BRANCH IF /NODIRECTORY
 5C       40 AB     7E  03BA              MOVAQ   PRC_Q_KEYPAD(R11),AP               ;GET ADDRESS OF KEYPAD SYMBOL TABLE
             5C     DD  03BE              PUSHL   AP                                 ;SAVE R6
             56     7C  03C0              CLRQ    R6                                 ;SET INITIAL STATE DESCRIPTOR
       5C    6C     D0  03C2     210$:    MOVL    (AP),AP                            ;GET ADDRESS OF NEXT ENTRY
       6E    5C     D1  03C5              CMPL    AP,(SP)                            ;END OF TABLE?
             2A     13  03C8              BEQL    230$                               ;IF EQL, THEN DONE
    51    DC AC     9A  03CA              MOVZBL  SYM_T_SYMBOL(AP),R1                ;GET LENGTH OF SYMBOL
 52    0F AC41      9E  03CE              MOVAB   SYM_T_SYMBOL+3(AP)[R1],R2          ;GET ADDRESS OF IF_STATE LENGTH
       51    82     9A  03D3              MOVZBL  (R2)+,R1                           ;GET IF_STATE LENGTH
       56    51     D1  03D6              CMPL    R1,R6                              ;STATES MATCH
             11     12  03D9              BNEQ    220$                               ;NO, LIST IT
       7E    51     7D  03DB              MOVQ    R1,-(SP)                           ;SAVE STATE DESCRIPTOR
 67    62    51     29  03DE              CMPC3   R1,(R2),(R7)                       ;STATES MATCH?
             05     12  03E2              BNEQ    215$                               ;NO, THEN OUTPUT THE STATE
          51    8E   7D  03E4              MOVQ    (SP)+,R1                        ;RESTORE STATE DESCRIPTOR
                D9   11  03E7              BRB     210$                            ;YES, THEN SKIP
          51    8E   7D  03E9     215$:    MOVQ    (SP)+,R1                        ;RESTORE STATE DESCRIPTOR
          56    51   7D  03EC     220$:    MOVQ    R1,R6                           ;SAVE NEW STATE
             FC0E'   30  03EF              BSBW    DCL$MSGOUT                      ;OUTPUT THE STATE
                CE   11  03F2              BRB     210$                            ;GET NEXT STATE
                8E   D5  03F4     230$:    TSTL    (SP)+                           ;RESTORE THE STACK
                49   11  03F6              BRB     90$                             ;EXIT
                         03F8
                         03F8     ;
                         03F8     ; SET KEYPAD STATE.
                         03F8     ;
                         03F8              ASSUME  PTR_K_COMMA NE 0
   BA AA   04 AE     D0  03F8     21$:     MOVL    4(SP),WRK_L_RSLNXT(R10)         ;RESET FOR FIRST STATE VALUE
                0A   13  03FD              BEQL    23$                             ;SKIP IF NONE
             FBFE'   30  03FF     22$:     BSBW    DCL$GETDVAL                     ;GET NEXT DESCRIPTOR VALUE
       04 AE    54   D0  0402              MOVL    R4,4(SP)                        ;SAVE THE TERMINATOR
              038B   30  0406              BSBW    DCL$ALLOC_STATE                 ;SET NEW STATE
                         0409     ;
                         0409     ; DETERMINE WHETHER DISPLAYING ONE SYMBOL OR /ALL SYMBOLS.
                         0409     ;
          51 6E      E9  0409     23$:     BLBC    (SP),40$                        ;BRANCH IF /ALL
                         040C
                         040C     ;
                         040C     ; DISPLAY SPECIFIED SINGLE SYMBOL VALUE.
                         040C     ;
          51    56   7D  040C              MOVQ    R6,R1                           ;GET DESCRIPTOR OF SYMBOL NAME
       50    08 AE   D0  040F              MOVL    8(SP),R0                        ;GET /LOG FLAG
              023B   30  0413              BSBW    DCL$SYNONYM                     ;CHECK FOR SYNONYM KEY NAMES
          56    51   7D  0416              MOVQ    R1,R6                           ;SAVE DESCRIPTOR IN CASE OF UNDKEY
              02C2   30  0419              BSBW    DCL$FIND_KEYPAD                 ;FIND SPECIFIED SYMBOL
          11 50      E8  041C              BLBS    R0,35$                          ;BRANCH IF FOUND
                         041F
                         041F     ;
                         041F     ; OUTPUT WARNING MESSAGE.
                         041F     ;
       05 6E    02   E3  041F              BBCS    #2,(SP),33$                     ;SKIP BLANK LINE IF FIRST LINE
                51   7C  0423              CLRQ    R1                              ;SET NULL STRING
             FBD8'   30  0425              BSBW    DCL$MSGOUT                      ;OUTPUT THE BLANK LINE
          6E    08   C8  0428     33$:     BISL    #8,(SP)                         ;SET UNDEFINED SYMBOL FLAG
              FED0   30  042B              BSBW    UNDKEY                          ;OUTPUT UNDEFINED KEY MSG
                06   11  042E              BRB     38$                             ;GET NEXT TABLE
                         0430
                         0430     ;
                         0430     ; DISPLAY THE SYMBOL DATA.
                         0430     ;
              0150   30  0430     35$:     BSBW    DISPHDR                         ;DISPLAY KEYPAD TABLE NAME
              0060   30  0433              BSBW    DISPSYMB                        ;DISPLAY THE SYMBOL DATA
       04 AE    05   D1  0436     38$:     CMPL    #PTR_K_COMMA,4(SP)              ;TERMINATOR A COMMA?
                05   12  043A              BNEQ    90$                             ;NO, TIME TO EXIT
             FBC1'   30  043C              BSBW    DCL$LOCKED_STATE                ;YES, RESTORE LOCKED KEY STATE
                         043F                                                      ;  BEFORE GETTING NEXT STATE
                BE   11  043F              BRB     22$                             ;GET NEXT STATE
                         0441
                         0441     ;
                         0441     ; RESTORE KEYPAD STATE, SET NORMAL STATUS, AND EXIT.
                         0441     ;
```

```
                              0441  90$:    STATUS  NORMAL                          ;ASSUME SUCCESSFUL COMPLETION
                51   8E   D0  0448          MOVL    (SP)+,R1                        ;GET FLAGS
             07 51   03   E1  044B          BBC     #3,R1,95$                       ;BRANCH IF NO UNDEFINED SYMBOLS
       50   10038260 8F   D0  044F          MOVL    #CLI$_UNDKEY!STS$M_INHIB_MSG,R0 ;SET STATUS, INHIBIT RESIGNAL
                5E   08   C0  0456  95$:    ADDL    #8,SP                           ;RESTORE THE STACK
                  FBA4'   30  0459          BSBW    DCL$LOCKED_STATE                ;RESTORE KEY STATE
                          05  045C          RSB
                              045D
                              045D  ;
                              045D  ; DISPLAY ALL SYMBOL ENTRIES FOR THE SPECIFIED OR CURRENT STATE.
                              045D  ;
                   0123   30  045D  40$:    BSBW    DISPHDR                         ;DISPLAY KEYPAD TABLE NAME
             56   40 AB   7E  0460          MOVAQ   PRC_Q_KEYPAD(R11),R6            ;GET ADDRESS OF KEYPAD SYMBOL TABLE
                5C   56   D0  0464          MOVL    R6,AP                           ;COPY ADDRESS OF TABLE LISTHEAD
                              0467
                              0467  ;
                              0467  ; GET NEXT SYMBOL.
                              0467  ;
                56   66   D0  0467  50$:    MOVL    (R6),R6                         ;GET ADDRESS OF NEXT ENTRY
                5C   56   D1  046A          CMPL    R6,AP                           ;END OF TABLE?
                     C7   13  046D          BEQL    38$                             ;IF EQL YES
                              046F
                              046F  ;
                              046F  ; IF STATE DOES NOT MATCH, THEN SKIP THIS SYMBOL.
                              046F  ;
             54   0C A6   9E  046F          MOVAB   SYM_T_SYMBOL(R6),R4             ;GET ADDRESS OF SYMBOL NAME
                51   84   9A  0473          MOVZBL  (R4)+,R1                        ;GET LENGTH OF SYMBOL NAME
           54   02 A441   9E  0476          MOVAB   2(R4)[R1],R4                    ;GET ADDRESS OF IF STATE
                50   84   9A  047B          MOVZBL  (R4)+,R0                        ;GET IF STATE LENGTH
             52   48 AB   D0  047E          MOVL    PRC_L_CURRKEY(R11),R2           ;GET CURRENT STATE LENGTH/ADDRESS
                51   82   9A  0482          MOVZBL  (R2)+,R1                        ;
 64   50   00   62   51   2D  0485          CMPC5   R1,(R2),#0,R0,(R4)              ;STATES MATCH?
                     A9   19  048B  65$:    BLSS    38$                             ;NO, GET NEXT STATE
                     D8   14  048D          BGTR    50$                             ;NO, GET NEXT SYMBOL
                              048F
                              048F  ;
                              048F  ; STATE DID MATCH.  DISPLAY THE SYMBOL.
                              048F  ;
                53   56   D0  048F  70$:    MOVL    R6,R3                           ;SET ADDRESS OF SYMBOL
                     02   10  0492          BSBB    DISPSYMB                        ;FORMAT AND OUTPUT ENTRY
                     D1   11  0494          BRB     50$                             ;GET NEXT
```

```
                        0496  ;+
                        0496  ; DISPSYMB - DISPLAY THE VALUE AND ATTRIBUTES OF A GIVEN KEYPAD SYMBOL.
                        0496  ;
                        0496  ; INPUTS:
                        0496  ;
                        0496  ;         4(SP) = FLAGS LONGWORD - BIT 1 IS SET IF /BRIEF
                        0496  ;         R3 = ADDRESS OF SYMBOL TABLE ENTRY
                        0496  ;         R8 = ADDRESS OF SCRATCH BUFFER DESCRIPTOR.
                        0496  ;         R9 = ADDRESS OF SCRATCH STACK.
                        0496  ;-
                        0496  DISPSYMB:                                          ;FORMAT A SYMBOL
    50   04 AE     D0   0496          MOVL    4(SP),R0                    ;GET THE FLAGS
       02C0 8F     BB   049A          PUSHR   #^M<R6,R7,R9>               ;SAVE REGISTERS
            68     DD   049E          PUSHL   (R8)                        ;SAVE SCRATCH DESCR LENGTH
            58     DD   04A0          PUSHL   R8                          ;SAVE ADDR OF SCRATCH DESCR
    58      50     D0   04A2          MOVL    R0,R8                       ;COPY THE FLAGS
                        04A5
                        04A5  ;
                        04A5  ; GET AND SAVE DESCRIPTOR OF SYMBOL NAME.
                        04A5  ;
    52   0C A3     9E   04A5          MOVAB   SYM_T_SYMBOL(R3),R2         ;POINT TO SYMBOL NAME
      51    82     9A   04A9          MOVZBL  (R2)+,R1                    ;GET NAME LENGTH
      79    51     7D   04AC          MOVQ    R1,-(R9)                    ;BUILD NAME DESCRIPTOR
      57    59     D0   04AF          MOVL    R9,R7                       ;COPY SCRATCH STACK POINTER
                        04B2
                        04B2  ;
                        04B2  ; GET AND SAVE DESCRIPTOR OF SYMBOL VALUE.
                        04B2  ;
  52   02 A241     9E   04B2          MOVAB   2(R2)[R1],R2                ;GET ADDRESS OF IF_STATE LENGTH
      51    82     9A   04B7          MOVZBL  (R2)+,R1                    ;GET LENGTH OF IF_STATE
    52    6241     9E   04BA          MOVAB   (R2)[R1],R2                 ;GET ADDRESS OF SYMBOL VALUE LENGTH
      51    82     3C   04BE          MOVZWL  (R2)+,R1                    ;GET LENGTH/ADDRESS OF VALUE
      79    51     7D   04C1          MOVQ    R1,-(R9)                    ;SAVE VALUE DESCRIPTOR
      56    59     D0   04C4          MOVL    R9,R6                       ;COPY SCRATCH STACK POINTER
                        04C7
                        04C7  ;
                        04C7  ; GET AND SAVE DESCRIPTOR OF SET_STATE STRING.
                        04C7  ;
    52    6241     9E   04C7          MOVAB   (R2)[R1],R2                 ;GET ADDRESS OF SET_STATE LENGTH
      51    82     9A   04CB          MOVZBL  (R2)+,R1                    ;GET LENGTH/ADDRESS OF STATE
   02 58    01     E1   04CE          BBC     #1,R8,10$                   ;SKIP IF /NOBRIEF
            51     D4   04D2          CLRL    R1                          ;PUSH NULL STRING
      79    51     7D   04D4  10$:    MOVQ    R1,-(R9)                    ;SAVE STATE DESCRIPTOR
      55    59     D0   04D7          MOVL    R9,R5                       ;COPY SCRATCH STACK POINTER
                        04DA
                        04DA  ;
                        04DA  ; CREATE AND SAVE DESCRIPTOR OF ASCIC FAO STRING.  OUTPUT WILL LOOK LIKE:
                        04DA  ;
                        04DA  ;         symbol = "value"        (ECHO,TERMINATE,ERASE,LOCK,STATE=state)
                        04DA  ;
  52   FB48 CF     9E   04DA          MOVAB   FULLFAO,R2                  ;ASSUME FULL DISPLAY
   05 58    01     E1   04DF          BBC     #1,R8,20$                   ;SKIP IF /NOBRIEF
  52   FB31 CF     9E   04E3          MOVAB   BRIEFFAO,R2                 ;SET BRIEF DISPLAY
      51    82     9A   04E8  20$:    MOVZBL  (R2)+,R1                    ;MAKE INTO DESCRIPTOR
      79    51     7D   04EB          MOVQ    R1,-(R9)                    ;AND PUSH ONTO STACK
      54    59     D0   04EE          MOVL    R9,R4                       ;COPY SCRATCH STACK POINTER
   21 58    01     E0   04F1          BBS     #1,R8,30$                   ;SKIP IF /BRIEF
```

```
                       04F5
                       04F5  ;
                       04F5  ; CREATE FAO PARAMETER LIST.  ASSUME NO ATTRIBUTES.
                       04F5  ;
       79   55    D0   04F5          MOVL    R5,-(R9)                  ;SET ADDR OF STATE DESCR
  79   FB74 CF    9E   04F8          MOVAB   NULL,-(R9)                ;ASSUME STATE FLAG NOT SET
  79   FB6F CF    9E   04FD          MOVAB   NULL,-(R9)                ;ASSUME STATE FLAG NOT SET
  79   FB6A CF    9E   0502          MOVAB   NULL,-(R9)                ;ASSUME LOCK FLAG SET
  79   FB65 CF    9E   0507          MOVAB   NULL,-(R9)                ;ASSUME ERASE FLAG SET
  79   FB60 CF    9E   050C          MOVAB   NULL,-(R9)                ;ASSUME TERMINATE FLAG SET
  79   FB5B CF    9E   0511          MOVAB   NULL,-(R9)                ;ASSUME ECHO FLAG SET
       79   56    D0   0516  30$:    MOVL    R6,-(R9)                  ;SET ADDR OF VALUE DESCR
       79   57    D0   0519          MOVL    R7,-(R9)                  ;SET ADDR OF NAME DESCR
       57   59    D0   051C          MOVL    R9,R7                     ;SAVE ADDRESS OF PARAMETER LIST
                       051F
                       051F  ;
                       051F  ; NOW RESET FAO ARGUMENTS FOR ANY ATTRIBUTES THAT WERE ABSENT.
                       051F  ;
    03 58   01    E1   051F          BBC     #1,R8,40$                 ;SKIP IF /NOBRIEF
          003D    31   0523          BRW     90$                       ;BRANCH IF /BRIEF
                       0526
            00    E0   0526  40$:    BBS     #SYM_V_ECHO,-             ;IS ECHO SET?
       06 0B A3        0528                  SYM_B_FLAGS(R3),50$       ;
08 A9  FB37 CF    9E   052B          MOVAB   NO,8(R9)                  ;
                       0531
            01    E0   0531  50$:    BBS     #SYM_V_TERMINATE,-        ;IS TERMINATE SET?
       06 0B A3        0533                  SYM_B_FLAGS(R3),60$       ;
0C A9  FB2C CF    9E   0536          MOVAB   NO,12(R9)                 ;
                       053C
            04    E0   053C  60$:    BBS     #SYM_V_ERASE,-            ;IS ERASE SET?
       06 0B A3        053E                  SYM_B_FLAGS(R3),70$       ;
10 A9  FB21 CF    9E   0541          MOVAB   NO,16(R9)                 ;
                       0547
            03    E0   0547  70$:    BBS     #SYM_V_LOCK,-             ;IS LOCK SET?
       06 0B A3        0549                  SYM_B_FLAGS(R3),80$       ;
14 A9  FB16 CF    9E   054C          MOVAB   NO,20(R9)                 ;
                       0552
            02    E1   0552  80$:    BBC     #SYM_V_STATE,-            ;IS STATE SET?
       0C 0B A3        0554                  SYM_B_FLAGS(R3),90$       ;
18 A9  FB16 CF    9E   0557          MOVAB   COMMA,24(R9)              ;
1C A9  FB08 CF    9E   055D          MOVAB   STATE,28(R9)              ;
                       0563
                       0563  ;
                       0563  ; FORMAT AND OUTPUT THE MESSAGE
                       0563  ;
          58 8ED0      0563  90$:    POPL    R8                        ;RESTORE SCRATCH DESCRIPTOR
                       0566          $FAOL_S (R4),(R8),(R8),(R7)       ;FORMAT OUTPUT MESSAGE
       51   68    7D   0575          MOVQ    (R8),R1                   ;GET OUTPUT MESSAGE PARAMETERS
          FA85'   30   0578          BSBW    DCL$MSGOUT                ;OUTPUT THE MESSAGE
          68 8ED0      057B          POPL    (R8)                      ;RESTORE SCRATCH DESCR LENGTH
       02C0 8F    BA   057E          POPR    #^M<R6,R7,R9>             ;RESTORE REGISTERS
            05         0582          RSB                               ;RETURN
```

```
                    0583  ;+
                    0583  ; DISPHDR - DISPLAY A KEYPAD TABLE HEADER
                    0583  ;
                    0583  ; INPUTS:
                    0583  ;
                    0583  ;       4(SP) = FLAGS LONGWORD - BIT 2 IS CLEAR IF FIRST TABLE
                    0583  ;       R8 = ADDRESS OF SCRATCH BUFFER DESCRIPTOR.
                    0583  ;       R9 = ADDRESS OF SCRATCH STACK.
                    0583  ;       PRC_L_CURRKEY(R11) = ADDDRESS OF ASCIC TABLE NAME
                    0583  ;-
                    0583  DISPHDR:                                        ;DISPLAY KEYPAD TABLE HEADER
05 04 AE    02  E3  0583          BBCS    #2,4(SP),10$                    ;SKIP BLANK LINE IF FIRST HEADER
            51  7C  0588          CLRQ    R1                              ;SET NULL STRING
         FA73'  30  058A          BSBW    DCL$MSGOUT                      ;OUTPUT THE BLANK LINE
            68  DD  058D  10$:    PUSHL   (R8)                            ;SAVE BUFFER SIZE
 52    FA6D CF  9E  058F          MOVAB   SHOWHDR,R2                      ;GET ADDRESS OF ASCIC FAO STRING
      51    82  9A  0594          MOVZBL  (R2)+,R1                        ;MAKE INTO DESCRIPTOR
      7E    51  7D  0597          MOVQ    R1,-(SP)                        ;AND PUSH ONTO STACK
      50    5E  D0  059A          MOVL    SP,R0                           ;SAVE ITS ADDRESS
   51    48 AB  9E  059D          MOVAB   PRC_L_CURRKEY(R11),R1           ;GET ADDRESS OF ASCIC STATE
                    05A1          $FAO_S  (R0),(R8),(R8),(R1)             ;FORMAT OUTPUT MESSAGE
      51    68  7D  05B0          MOVQ    (R8),R1                         ;GET OUTPUT MESSAGE PARAMETERS
         FA4A'  30  05B3          BSBW    DCL$MSGOUT                      ;OUTPUT THE MESSAGE
      5E    08  C0  05B6          ADDL    #8,SP                           ;RESTORE THE STACK
      68    8E  D0  05B9          MOVL    (SP)+,(R8)                      ;RESTORE BUFFER SIZE
                05  05BC          RSB                                     ;RETURN
```

```
                 05BD          .SBTTL  ALLOCATE AND INSERT ENTRY IN KEYPAD SYMBOL TABLE
                 05BD    ;+
                 05BD    ; DCL$ALLOCKEY - ALLOCATE AND INSERT ENTRY IN KEYPAD SYMBOL TABLE
                 05BD    ;
                 05BD    ; THIS ROUTINE IS CALLED TO ALLOCATE AND INSERT AN ENTRY IN THE KEYPAD
                 05BD    ; SYMBOL TABLE.
                 05BD    ;
                 05BD    ; INPUTS:
                 05BD    ;
                 05BD    ;       R6 = KEYPAD FLAGS
                 05BD    ;       R9 = ADDRESS OF BUFFER FORMATTED AS FOLLOWS
                 05BD    ;
                 05BD    ;               (R9) = DESCRIPTOR OF SYMBOL VALUE
                 05BD    ;               8(R9) = DESCRIPTOR OF SYMBOL NAME
                 05BD    ;               16(R9) = DESCRIPTOR OF SET_STATE NAME
                 05BD    ;
                 05BD    ;       R11 = ADDRESS OF PROCESS WORK AREA
                 05BD    ;       PRC_L_CURRKEY = STATE IN WHICH KEY IS TO BE ALLOCATED
                 05BD    ;
                 05BD    ; OUTPUTS:
                 05BD    ;
                 05BD    ;       THE KEYPAD TABLE IS SEARCHED FOR THE SPECIFIED ENTRY, AND IF FOUND,
                 05BD    ;       THE OLD ENTRY IS DEALLOCATED. A SYMBOL TABLE ENTRY IS THEN ALLOCATED,
                 05BD    ;       FILLED WITH THE SYMBOL, VALUE, AND STATE INFORMATION, AND THEN
                 05BD    ;       INSERTED IN THE SYMBOL TABLE.
                 05BD    ;
                 05BD    ;       R0 LOW BIT CLEAR INDICATES ALLOCATION FAILURE WITH CLI$_SYMOVF.
                 05BD    ;       R0 LOW BIT SET INDICATES SUCCESSFUL COMPLETION.
                 05BD    ;
                 05BD    ;               R1,R2,R3,R4,R5 ARE DESTROYED.
                 05BD    ;-
                 05BD
                 05BD    DCL$ALLOCKEY::
                 05BD          DISABLE                                 ;DISABLE CTRL/Y'S
       8E   7C   05C3          CLRQ    (SP)+                           ;REMOVE RETURN INFO FROM STACK
                 05C5
                 05C5    ;
                 05C5    ; SEARCH FOR PREVIOUS DEFINITION OF IDENTICAL SYMBOL.  IF FOUND, THEN
                 05C5    ; DEALLOCATE IT.  FIND THE SPOT IN THE LINKED LIST TO INSERT THE NEW SYMBOL
                 05C5    ; AT.
                 05C5    ;
51    08 A9     7D   05C5          MOVQ    8(R9),R1                        ;SET SYMBOL NAME DESCRIPTOR
       0112     30   05C9          BSBW    DCL$FIND_KEYPAD                 ;SEARCH FOR SYMBOL
      0C 50     E9   05CC          BLBC    R0,10$                          ;IF LBC SEARCH FAILURE
   7E    51     7D   05CF          MOVQ    R1,-(SP)                        ;SAVE R1/R2
      FA2B'     30   05D2          BSBW    DCL$DEALLOCSYM                  ;DEALLOCATE KEYPAD ENTRY
   51    8E     7D   05D5          MOVQ    (SP)+,R1                        ;RESTORE R1/R2
       0103     30   05D8          BSBW    DCL$FIND_KEYPAD                 ;SEARCH FOR SYMBOL
                 05DB
                 05DB    ;
                 05DB    ; CALCULATE SIZE OF NEW SYMBOL AND ALLOCATE IT.
                 05DB    ;
51    48 AB     D0   05DB    10$:  MOVL    PRC_L_CURRKEY(R11),R1           ;GET ADDR OF ASCIC IF_STATE
   51    61     9A   05DF          MOVZBL  (R1),R1                         ;GET LENGTH OF IF_STATE
51    10 A9     C0   05E2          ADDL    16(R9),R1                       ;ADD IN SET_STATE_LENGTH
   51    69     C0   05E6          ADDL    (R9),R1                         ;ADD IN VALUE LENGTH
   51    04     C0   05E9          ADDL    #4,R1                           ;ADD IN SIZE OF LENGTH FIELDS
```

```
             51    DD  05EC          PUSHL   R1                              ;SAVE FOR FUTURE USE
     51   08 A9    C0  05EE          ADDL    8(R9),R1                        ;ADD IN META-KEY LENGTH
        51   0F    C0  05F2          ADDL    #SYM_T_SYMBOL+3,R1              ;ADD IN FIXED OVERHEAD
             53    DD  05F5          PUSHL   R3                              ;SAVE SYMBOL TABLE PTR
          FA06'    30  05F7          BSBW    DCL$ALLDYNMEM                   ;ALLOCATE DYNAMIC MEMORY
             53  8ED0  05FA          POPL    R3                              ;RESTORE SYMBOL TABLE PTR
          47 50    E9  05FD          BLBC    R0,90$                          ;IF LBC ALLOCATION FAILURE
                       0600
                       0600    ;
                       0600    ; INITIALIZE THE STATICALLY PLACED FIELDS AND INSERT IT IN THE LINKED LIST.
                       0600    ;
     08 A2   51    B0  0600          MOVW    R1,SYM_W_SIZE(R2)               ;SET SIZE OF ALLOCATED BLOCK
     0B A2   56    90  0604          MOVB    R6,SYM_B_FLAGS(R2)              ;SET KEYPAD FLAGS
     0A A2   04    90  0608          MOVB    #SYM_K_KEYPAD,SYM_B_TYPE(R2)    ;SET KEYPAD VALUE TYPE
     04 B3   62    0E  060C          INSQUE  SYM_L_FL(R2),@SYM_L_BL(R3)      ;INSERT ENTRY IN SYMBOL TABLE
                       0610
                       0610    ;
                       0610    ; INITIALIZE THE DYNAMICALLY PLACED ASCIC FIELDS.
                       0610    ;
     53   08 A9    7D  0610          MOVQ    8(R9),R3                        ;GET SYMBOL NAME
        0C A2  53  90  0614          MOVB    R3,SYM_T_SYMBOL(R2)             ;INSERT LENGTH OF SYMBOL
OD A2   64   53    28  0618          MOVC    R3,(R4),SYM_T_SYMBOL+1(R2)      ;INSERT SYMBOL NAME
                       061D
        83   8E    F7  061D          CVTLW   (SP)+,(R3)+                     ;INSERT LENGTH OF FOLLOWING
                       0620
     52   48 AB    D0  0620          MOVL    PRC_L_CURRKEY(R11),R2           ;GET CURRENT STATE LENGTH/ADDRESS
        51   82    9A  0624          MOVZBL  (R2)+,R1                        ;GET LENGTH OF STRING VALUE
        83   51    90  0627          MOVB    R1,(R3)+                        ;INSERT LENGTH OF STRING VALUE
     63 62   51    28  062A          MOVC    R1,(R2),(R3)                    ;INSERT STRING VALUE
                       062E
        51   69    7D  062E          MOVQ    (R9),R1                         ;GET SYMBOL VALUE
        83   51    B0  0631          MOVW    R1,(R3)+                        ;INSERT LENGTH OF STRING VALUE
     63 62   51    28  0634          MOVC    R1,(R2),(R3)                    ;INSERT STRING VALUE
                       0638
     51   10 A9    7D  0638          MOVQ    16(R9),R1                       ;GET SET STATE VALUE
        83   51    90  063C          MOVB    R1,(R3)+                        ;INSERT LENGTH OF STRING VALUE
     63 62   51    28  063F          MOVC    R1,(R2),(R3)                    ;INSERT STRING VALUE
                       0643
        50   01    D0  0643          MOVL    #1,R0                           ;SET SUCCESS INDICATOR
                   05  0646          RSB                                     ;
                       0647
                       0647    ;
                       0647    ; RETURN SYMBOL TABLE OVERFLOW STATUS.
                       0647    ;
             8E    D5  0647    90$:  TSTL    (SP)+                           ;RESTORE THE STACK
                       0649          STATUS  SYMOVF                          ;SET SYMBOL TABLE OVERFLOW STATUS
                   05  0650          RSB
```

```
                                 0651          .SBTTL  CHECK FOR SYNONYM KEY NAMES
                                 0651  ;+
                                 0651  ; DCL$SYNONYM - CHECK FOR SYNONYM KEY NAMES
                                 0651  ;
                                 0651  ; THIS ROUTINE IS CALLED TO DETERMINE WHETHER OR NOT THE KEY NAME INPUT
                                 0651  ; HAS A SYNONYM NAME. IF SO, IT TRANSLATES THE KEY NAME TO A COMMON KEY NAME
                                 0651  ; FOR THAT PARTICULAR KEY, WHICH IS THEN USED IN CONSTRUCTING THE KEYPAD
                                 0651  ; SYMBOL TABLE. IF /LOG IS SPECIFIED IN THE COMMAND LINE, A CONVERSION MESSAGE
                                 0651  ; IS OUTPUT INDICATING WHAT THE SYNONYM KEY WAS CHANGED TO IN THE KEYPAD
                                 0651  ; SYMBOL TABLE.
                                 0651  ;
                                 0651  ;
                                 0651  ; INPUTS:
                                 0651  ;
                                 0651  ;       R0 = /LOG FLAG (LBS = /LOG, LBC = /NOLOG)
                                 0651  ;       R1 = LENGTH OF ENTERED KEY NAME
                                 0651  ;       R2 = ADDRESS OF ENTERED KEY NAME
                                 0651  ;
                                 0651  ; OUTPUTS:
                                 0651  ;
                                 0651  ;       IF SYNONYM FOUND:
                                 0651  ;
                                 0651  ;       R1 = LENGTH OF TRANSLATED KEY NAME
                                 0651  ;       R2 = ADDR. OF TRANSLATED KEY NAME
                                 0651  ;
                                 0651  ;
                                 0651  ;       IF SYNONYM NOT FOUND:
                                 0651  ;
                                 0651  ;       R1  UNCHANGED
                                 0651  ;       R2  UNCHANGED
                                 0651  ;
                                 0651  ;-
                                 0651
                                 0651 DCL$SYNONYM::
                 7E   56   7D    0651          MOVQ    R6,-(SP)                ;SAVE WORK REGISTERS
                 7E   50   D0    0654          MOVL    R0,-(SP)                ;SAVE /LOG FLAG
            56 FA18   CF   9E    0657          MOVAB   SYNNAME_TAB,R6          ;GET ADDR OF SYNONYM TABLE
                                 065C
                 57   66   9A    065C 10$:     MOVZBL  (R6),R7                 ;GET LENGTH OF THIS ENTRY
                      5A   13    065F          BEQL    100$                    ;EXIT IF NO MATCHING ENTRY FOUND
                 57   51   B1    0661          CMPW    R1,R7                   ;DOES THE LENGTH MATCH THIS ENTRY?
                      0F   12    0664          BNEQ    40$                     ;NO, SKIP TO NEXT ENTRY IN TABLE
                 7E   51   7D    0666          MOVQ    R1,-(SP)                ;SAVE POINTERS
  01 A6    57   00   62   51   2D    0669          CMPC5   R1,(R2),#0,R7,1(R6)     ;IS THERE A MATCH ON THIS ENTRY?
                      0A   13    0670          BEQL    50$                     ;YES, GET NEW KEY NAME FROM TRANSLATION TABL
                 51   8E   7D    0672          MOVQ    (SP)+,R1                ;RESTORE POINTERS
                                 0675
            56 03 A647     9E    0675 40$:     MOVAB   3(R6)[R7],R6            ;MOVE TO NEXT ENTRY IN SYNONYM TABLE
                      E0   11    067A          BRB     10$
                                 067C  ;
                                 067C  ; HAVE FOUND A MATCH IN SYNONYM TABLE. GET ACTUAL KEY NAME FROM TRANSLATION TABLE.
                                 067C  ;
                 51   8E   7D    067C 50$:     MOVQ    (SP)+,R1                ;RESTORE POINTERS
            56 01 A647     9E    067F          MOVAB   1(R6)[R7],R6            ;GET ADDR. OF OFFSET INTO TRANS. TABLE
                 56   66   3C    0684          MOVZWL  (R6),R6                 ;GET ACTUAL OFFSET
        56  FA2B CF46      9E    0687          MOVAB   SYNDEF_TAB[R6],R6       ;GET ADDR. OF COMMON KEY NAME STRING
                 53   51   7D    068D          MOVQ    R1,R3                   ;SAVE ENTERED KEY NAME FOR CONV. MESSAGE.
                 51   86   9A  0690   977          MOVZBL  (R6)+,R1                   ;FORM DESCRIPTOR FOR TRANS. NAME
                 52   66   9E  0693   978          MOVAB   (R6),R2                    ;
                               0696   979 ;
                               0696   980 ; OUTPUT CONVERSION MESSAGE IF /LOG SPECIFIED
                               0696   981 ;
                 22 6E     E9  0696   982          BLBC    (SP),100$                  ;SKIP MESSAGE IF /NOLOG
                 5E   10   C2  0699   983          SUBL    #16,SP                     ;MAKE A TEMPORARY SCRATCH BUFFER
                 6E   51   7D  069C   984          MOVQ    R1,(SP)                    ;GET TRANSLATED KEY NAME
              08 AE   53   7D  069F   985          MOVQ    R3,8(SP)                   ;GET ENTERED KEY NAME
                      6E   9F  06A3   986          PUSHAB  (SP)                       ;SET ADDR. OF TRANS. KEY NAME
                   0C AE   9F  06A5   987          PUSHAB  12(SP)                     ;SET ADDR. OF ENTERED KEY NAME
                 51   02   D0  06A8   988          MOVL    #2,R1                      ;SET ARGUMENT COUNTER
       50   0003DE33 8F    D0  06AB   989          MOVL    #CLI$_KEYCNV,R0            ;SET CONVERTED KEY STATUS
                   F94B'   30  06B2   990          BSBW    DCL$FORMMSG                ;OUTPUT CONVERSION MESSAGE
                 51   6E   7D  06B5   991          MOVQ    (SP),R1                    ;RESTORE TRANSLATED  DESCRIPTOR
                 5E   10   C0  06B8   992          ADDL    #16,SP                     ;REMOVE TEMPORARY SCRATCH BUFFER
                               06BB   993
                      8E   D5  06BB   994 100$:    TSTL    (SP)+                      ;REMOVE /LOG FLAG
                 56   8E   7D  06BD   995          MOVQ    (SP)+,R6                   ;RESTORE WORK REGISTERS
                           05  06C0   996          RSB                                ;EXIT
```

```
                        06C1          .SBTTL  SEARCH FOR SYMBOL ENTRY IN KEYPAD SYMBOL TABLE
                        06C1  ;+
                        06C1  ; DCL$SEARCH_KEYPAD - SEARCH FOR SYMBOL ENTRY IN KEYPAD SYMBOL TABLE
                        06C1  ;
                        06C1  ; THIS ROUTINE IS CALLED TO SEARCH THE KEYPAD SYMBOL TABLE FOR AN ENTRY.
                        06C1  ;
                        06C1  ; INPUTS:
                        06C1  ;
                        06C1  ;       R11 = ADDRESS OF PROCESS WORK AREA
                        06C1  ;
                        06C1  ;       R1 = LENGTH OF SYMBOL.
                        06C1  ;       R2 = ADDRESS OF SYMBOL.
                        06C1  ;
                        06C1  ; OUTPUTS:
                        06C1  ;
                        06C1  ;       R0 = STATUS
                        06C1  ;       R1/R2 = QUADWORD DESCRIBING SYMBOL VALUE:
                        06C1  ;               IF R2 NONZERO, QUADWORD IS A STRING DESCRIPTOR
                        06C1  ;               IF R2 ZERO, R1 IS A BINARY LONGWORD VALUE
                        06C1  ;       R3 = ADDRESS OF SYMBOL ENTRY
                        06C1  ;       R4 = KEYPAD FLAGS
                        06C1  ;
                        06C1  ;-
                        06C1
                        06C1  DCL$SEARCH_KEYPAD::                        ;SEARCH FOR SYMBOL ENTRY IN KEYPAD TABLE
             1B     10  06C1          BSBB    DCL$FIND_KEYPAD             ;SEARCH KEYPAD SYMBOL TABLE FOR ENTRY
          17 50     E9  06C3          BLBC    R0,10$                      ;IF LBC NO MATCH FOUND
                        06C6
    51     0C A3    9A  06C6          MOVZBL  SYM_T_SYMBOL(R3),R1         ;GET LENGTH OF SYMBOL
 52      0F A341    9E  06CA          MOVAB   SYM_T_SYMBOL+3(R3)[R1],R2  ;GET ADDRESS OF IF_STATE LENGTH
        51    82    9A  06CF          MOVZBL  (R2)+,R1                    ;GET LENGTH OF IF STATE
    52      6241    9E  06D2          MOVAB   (R2)[R1],R2                 ;GET ADDRESS OF VALUE LENGTH
        51    82    3C  06D6          MOVZWL  (R2)+,R1                    ;GET LENGTH OF VALUE
  54     0B A3      9A  06D9          MOVZBL  SYM_B_FLAGS(R3),R4          ;GET KEYPAD FLAGS
                        06DD
                    05  06DD  10$:    RSB                                 ;
```

```
                            06DE          .SBTTL  SEARCH KEYPAD SYMBOL TABLE FOR ENTRY
                            06DE    ;+
                            06DE    ; DCL$FIND_KEYPAD - SEARCH KEYPAD SYMBOL TABLE FOR ENTRY
                            06DE    ;
                            06DE    ; THIS ROUTINE IS CALLED TO SEARCH THE KEYPAD SYMBOL TABLE FOR AN ENTRY.
                            06DE    ; ONLY DEFINITIONS FOR THE PRC_L_CURRKEY STATE ARE CHECKED.
                            06DE    ;
                            06DE    ; INPUTS:
                            06DE    ;
                            06DE    ;       R1 = LENGTH OF SYMBOL NAME.
                            06DE    ;       R2 = ADDRESS OF SYMBOL NAME.
                            06DE    ;       R11 = ADDRESS OF PRC DATA STRUCTURE
                            06DE    ;       PRC_L_CURRKEY = CURRENT KEY STATE
                            06DE    ;
                            06DE    ; OUTPUTS:
                            06DE    ;
                            06DE    ;       R0 LOW BIT CLEAR INDICATES SEARCH FAILURE.
                            06DE    ;
                            06DE    ;               R1 = LENGTH OF SYMBOL NAME.
                            06DE    ;               R2 = ADDRESS OF SYMBOL NAME.
                            06DE    ;               R3 = ADDRESS OF NEXT GREATEST SYMBOL ENTRY.
                            06DE    ;               R4 ARE DESTROYED.
                            06DE    ;
                            06DE    ;       R0 LOW BIT SET INDICATES SYMBOL FOUND WITH:
                            06DE    ;
                            06DE    ;               R1 = LENGTH OF SYMBOL NAME.
                            06DE    ;               R2 = ADDRESS OF SYMBOL NAME.
                            06DE    ;               R3 = ADDRESS OF SYMBOL ENTRY.
                            06DE    ;               R4 IS DESTROYED
                            06DE    ;-
                            06DE
                            06DE    DCL$FIND_KEYPAD::                               ;SEARCH KEYPAD SYMBOL TABLE FOR ENTRY
                            06DE
                            06DE    ;
                            06DE    ; SET ADDRESS OF SYMBOL TABLE.
                            06DE    ;
           53   40 AB   9E  06DE            MOVAB   PRC_Q_KEYPAD(R11),R3    ;SET ADDRESS OF KEYPAD SYMBOL TABLE LISTHEAD
              50   53   D0  06E2            MOVL    R3,R0                   ;COPY ADDRESS OF SYMBOL TABLE LISTHEAD
                            06E5
                            06E5    ;
                            06E5    ; SEARCH FOR THE SPECIFIED SYMBOL.
                            06E5    ;
              53   63   D0  06E5    10$:    MOVL    SYM_L_FL(R3),R3         ;GET ADDRESS OF NEXT ENTRY
              53   50   D1  06E8            CMPL    R0,R3                   ;END OF TABLE?
                   38   13  06EB            BEQL    90$                     ;IF EQL YES
                            06ED
                            06ED    ;
                            06ED    ; CHECK THAT THE SYMBOL STATE MATCHES.
                            06ED    ;
                   0F   BB  06ED            PUSHR   #^M<R0,R1,R2,R3>        ;SAVE SEARCH PARAMETERS
           51   0C A3   9A  06EF            MOVZBL  SYM_T_SYMBOL(R3),R1     ;GET LENGTH OF SYMBOL
         54   0F A341   9E  06F3            MOVAB   SYM_T_SYMBOL+3(R3)[R1],R4 ;GET ADDRESS OF IF_STATE LENGTH
              50   84   9A  06F8            MOVZBL  (R4)+,R0                ;GET IF STATE LENGTH
           52   48 AB   D0  06FB            MOVL    PRC_L_CURRKEY(R11),R2   ;GET CURRENT STATE LENGTH/ADDRESS
              51   82   9A  06FF            MOVZBL  (R2)+,R1                ;
  64   50   00   62   51   2D  0702            CMPC5   R1,(R2),#0,R0,(R4)      ;STATES MATCH?
                   0F   BA  0708    20$:    POPR    #^M<R0,R1,R2,R3>        ;RESTORE SEARCH PARAMETERS
```

```
                        D9   14  070A  1092          BGTR    10$                        ;IF NEQ NO
                        17   19  070C  1093          BLSS    90$                        ;
                                 070E  1094
                                 070E  1095  ;
                                 070E  1096  ; CHECK THAT THE SYMBOL NAME MATCHES.
                                 070E  1097  ;
               54    0C A3   9E  070E  1098          MOVAB   SYM_T_SYMBOL(R3),R4        ;GET ADDRESS OF SYMBOL NAME
                        0F   BB  0712  1099          PUSHR   #^M<R0,R1,R2,R3>           ;SAVE SEARCH PARAMETERS
                     50 84   9A  0714  1100          MOVZBL  (R4)+,R0                   ;GET LENGTH OF SYMBOL NAME
     64    50    00  62 51   2D  0717  1101          CMPC5   R1,(R2),#0,R0,(R4)         ;SYMBOLS MATCH?
                        0F   BA  071D  1102          POPR    #^M<R0,R1,R2,R3>           ;RESTORE SEARCH PARAMETERS
                        C4   14  071F  1103          BGTR    10$                        ;IF NEQ NO
                        02   19  0721  1104          BLSS    90$                        ;
                                 0723  1105
                        50   D6  0723  1106          INCL    R0                         ;SET SUCCESS INDICATOR
                             05  0725  1107  90$:    RSB                                ;
                                 0726  1108
```

```
                    0726             .SBTTL  SET KEYPAD STATE
                    0726      ;+
                    0726      ; DCL$SETKEY - SET KEYPAD STATE
                    0726      ;
                    0726      ; THIS ROUTINE IS CALLED AS AN INTERNAL COMMAND TO EXECUTE THE SET KEYPAD
                    0726      ; COMMAND.
                    0726      ;
                    0726      ; INPUTS:
                    0726      ;
                    0726      ;        R8 = ADDRESS OF SCRATCH BUFFER DESCRIPTOR.
                    0726      ;        R9 = ADDRESS OF SCRATCH STACK.
                    0726      ;        R10 = BASE ADDRESS OF COMMAND WORK AREA.
                    0726      ;        R11 = BASE ADDRESS OF PROCESS WORK AREA.
                    0726      ;
                    0726      ; OUTPUTS:
                    0726      ;
                    0726      ;        THE SPECIFIED KEYPAD STATE BECOMES THE LOCKED CURRENT STATE.
                    0726      ;-
                    0726
                    0726      DCL$SETKEY::                                    ;SET KEYPAD SYMBOL TABLE STATE
      58   01   D0  0726             MOVL    #1,R8                           ;ASSUME /LOG
           56   7C  0729             CLRQ    R6                              ;INIT STATE NAME DESCRIPTOR
                    072B
                    072B      ;
                    072B      ; PROCESS THE TOKENS ON THE COMMAND LINE.
                    072B      ;
        FBD2'   30  072B      10$:   BSBW    DCL$GETDVAL                     ;GET NEXT DESCRIPTOR VALUE
      55   04   91  072E             CMPB    #PTR_K_ENDLINE,R5               ;END OF LINE?
           30   13  0731             BEQL    50$                             ;BRANCH IF SO
      55   03   91  0733             CMPB    #PTR_K_PARAMETR,R5              ;PARAMETER?
           F3   13  0736             BEQL    10$                             ;IGNORE IF SO
        F8C5'   30  0738             BSBW    DCL$GETNVAL                     ;GET QUALIFIER NUMBER
 00'8F     51   91  073B             CMPB    R1,#CLI$K_STKY_STAT             ;/STATE?
           08   13  073F             BEQL    20$                             ;YES, PROCESS IT
 00'8F     51   91  0741             CMPB    R1,#CLI$K_STKY_LOG              ;/LOG?
           10   13  0745             BEQL    30$                             ;YES, PROCESS IT
           E2   11  0747             BRB     10$                             ;NO, GET NEXT TOKEN
                    0749
           56   7C  0749      20$:   CLRQ    R6                              ;INIT STATE NAME DESCRIPTOR
 DC 53     00   E0  074B             BBS     #PTR_V_NEGATE-PTR_V_FLAGS,R3,10$ ;IGNORE IF NOT /STATE
        F8AE'   30  074F             BSBW    DCL$GETDVAL                     ;GET NEXT DESCRIPTOR VALUE
      56   51   7D  0752             MOVQ    R1,R6                           ;SAVE IT AWAY
           D4   11  0755             BRB     10$                             ;GET NEXT
                    0757
      58   01   C8  0757      30$:   BISL    #1,R8                           ;ASSUME /LOG
 CD   53   00   E1  075A             BBC     #PTR_V_NEGATE-PTR_V_FLAGS,R3,10$ ;IGNORE IF NOT /NOLOG
      58   01   CA  075E             BICL    #1,R8                           ;CLEAR FLAG
           C8   11  0761             BRB     10$                             ;GET NEXT
                    0763
                    0763      ;
                    0763      ; SET THE SPECIFIED STATE.
                    0763      ;
      51   56   7D  0763      50$:   MOVQ    R6,R1                           ;GET STATE DESCRIPTOR
           24   13  0766             BEQL    90$                             ;EXIT IF NONE SPECIFIED
           2A   10  0768             BSBB    DCL$ALLOC_STATE                 ;SET NEW STATE
        1F 50   E9  076A             BLBC    R0,90$                          ;EXIT IF ERROR
 50     4C AB   D0  076D             MOVL    PRC_L_LASTKEY(R11),R0           ;CLEAR LAST STATE
```

```
             0056  30  0771            BSBW    DCL$DEALLOC_STATE        ;
            48 AB  D0  0774            MOVL    PRC_L_CURRKEY(R11),-     ;COPY KEY DEFINITION
            4C AB      0777                    PRC_L_LASTKEY(R11)       ;
                       0779
                       0779      ;
                       0779      ; OUTPUT LOG MESSAGE IF REQUESTED.
                       0779      ;
            10 58  E9  0779            BLBC    R8,90$                   ;SKIP IF /NOLOG SPECIFIED
            48 AB  DD  077C            PUSHL   PRC_L_CURRKEY(R11)       ;SET ADDRESS OF ASCIC STATE NAME
         51    01  D0  077F            MOVL    #1,R1                    ;SET ARGUMENT COUNT
 50   0003DDD3 8F  D0  0782            MOVL    #CLI$_SETKEY,R0          ;SET STATUS
            F874'  30  0789            BSBW    DCL$FORMMSG              ;OUTPUT THE LOG MESSAGE
                       078C
                       078C      90$:  STATUS  NORMAL                   ;SET NORMAL SUCCESS STATUS
                   05  0793            RSB                              ;RETURN
                       0794
```

```
                      0794             .SBTTL  ALLOCATE AND INIT A KEYPAD STATE SYMBOL
                      0794      ;+
                      0794      ; DCL$ALLOC_STATE - ALLOCATE AND INIT A KEYPAD STATE SYMBOL
                      0794      ;
                      0794      ; THIS ROUTINE IS CALLED TO ALLOCATE AND INIT A KEYPAD STATE SYMBOL.
                      0794      ;
                      0794      ; INPUTS:
                      0794      ;
                      0794      ;      R1/R2 = DESCRIPTOR OF NEW KEYPAD STATE
                      0794      ;      R11 = ADDRESS OF PROCESS WORK AREA
                      0794      ;
                      0794      ; OUTPUTS:
                      0794      ;
                      0794      ;      PRC_L_CURRKEY = SET TO THE NEWLY ALLOCATED STATE SYMBOL
                      0794      ;
                      0794      ;      R0 LBC INDICATES SYMBOL TABLE OVERFLOW
                      0794      ;-
                      0794
                      0794      DCL$ALLOC_STATE::                                        ;ALLOCATE STATE SYMBOL
               3E  BB 0794             PUSHR   #^M<R1,R2,R3,R4,R5>                      ;SAVE R1-R5
                      0796             DISABLE                                          ;DISABLE CTRL/Y'S
          54   51  7D 079C             MOVQ    R1,R4                                    ;SAVE STATE DESCRIPTOR
               51  D6 079F             INCL    R1                                       ;ADD ROOM FOR BYTE COUNT
            F85C'  30 07A1             BSBW    DCL$ALLDYNMEM                            ;GET MEMORY TO SAVE STATE IN
            17 50  E9 07A4             BLBC    R0,90$                                   ;BRANCH IF NO ROOM FOR SYMBOL
       48 AB   52  D0 07A7             MOVL    R2,PRC_L_CURRKEY(R11)                    ;SET CURRENT KEY STATE
          82   54  90 07AB             MOVB    R4,(R2)+                                 ;MOVE THE STRING LENGTH
     62   65   54  28 07AE             MOVC3   R4,(R5),(R2)                             ;MOVE THE STRING
                      07B2             STATUS NORMAL                                    ;SET NORMAL STATUS
                      07B9             ENABLE                                           ;ENABLE CTRL/Y'S
               3E  BA 07BB             POPR    #^M<R1,R2,R3,R4,R5>                      ;RESTORE R1-R5
                   05 07BD             RSB                                              ;
                      07BE
                      07BE      90$:   ENABLE                                           ;ENABLE CTRL/Y'S
               3E  BA 07C0             POPR    #^M<R1,R2,R3,R4,R5>                      ;RESTORE R1-R5
                      07C2             STATUS  SYMOVF                                   ;
                   05 07C9             RSB                                              ;
                      07CA
```

```
                     07CA          .SBTTL  DEALLOCATE A KEYPAD STATE SYMBOL
                     07CA  ;+
                     07CA  ; DCL$DEALLOC_STATE - DEALLOCATE A KEYPAD STATE SYMBOL
                     07CA  ;
                     07CA  ; THIS ROUTINE IS CALLED TO DEALLOCATE A KEYPAD STATE SYMBOL.
                     07CA  ;
                     07CA  ; INPUTS:
                     07CA  ;
                     07CA  ;       R0 = ADDRESS OF ASCIC KEYPAD STATE
                     07CA  ;       R11 = ADDRESS OF PROCESS WORK AREA
                     07CA  ;
                     07CA  ; OUTPUTS:
                     07CA  ;
                     07CA  ;       NONE
                     07CA  ;-
                     07CA
                     07CA  DCL$DEALLOC_STATE::                                        ;DEALLOCATE STATE SYMBOL
                     07CA          DISABLE                                            ;DISABLE CTRL/Y'S
  7E    50     7D    07D0          MOVQ    R0,-(SP)                                   ;SAVE R0-R3
  7E    52     7D    07D3          MOVQ    R2,-(SP)                                   ;
  51    60     9A    07D6          MOVZBL  (R0),R1                                    ;GET LENGTH OF TEMPORARY STATE
        51     D6    07D9          INCL    R1                                         ;INCR TO INCLUDE BYTE COUNT
     F822'     30    07DB          BSBW    DCL$DEADYNMEM                              ;DEALLOCATE THE BLOCK
  52    8E     7D    07DE          MOVQ    (SP)+,R2                                   ;RESTORE R0-R3
  50    8E     7D    07E1          MOVQ    (SP)+,R0                                   ;
                     07E4          ENABLE                                             ;ENABLE CTRL/Y'S
               05    07E6          RSB                                                ;
                     07E7
                     07E7          .END
```

```
$$T2                 = 00000004                 NULL                 00000070 R     02
BRIEFFAO               00000018 R      02       PRC_B_CONTINUE       000000F3
CLI$K_DEFK_ECHO        ********    X   02       PRC_B_DEFRADIX       000000AE
CLI$K_DEFK_ERAS        ********    X   02       PRC_B_EXMDEPMOD      000000AD
CLI$K_DEFK_IF_S        ********    X   02       PRC_B_EXMDEPWID      000000AC
CLI$K_DEFK_LOCK        ********    X   02       PRC_B_EXONLYL        0000012D
CLI$K_DEFK_LOG         ********    X   02       PRC_B_FLAGS2         000000AF
CLI$K_DEFK_SET         ********    X   02       PRC_B_IMGFLAG        00000078
CLI$K_DEFK_TERM        ********    X   02       PRC_B_OUTFLAGS       0000012C
CLI$K_DELK_LOG         ********    X   02       PRC_B_PROMPTLEN      000000F0
CLI$K_DELK_STAT        ********    X   02       PRC_C_LENGTH         00000534
CLI$K_SHKY_BRIE        ********    X   02       PRC_G_COMMANDS       00000133
CLI$K_SHKY_DIRE        ********    X   02       PRC_G_PROMPT         000000F4
CLI$K_SHKY_FULL        ********    X   02       PRC_K_LENGTH         00000534
CLI$K_SHKY_LOG         ********    X   02       PRC_L_CURRKEY        00000048
CLI$K_SHKY_STAT        ********    X   02       PRC_L_EXMDEPADR      000000A8
CLI$K_STKY_LOG         ********    X   02       PRC_L_EXTARG         00000094
CLI$K_STKY_STAT        ********    X   02       PRC_L_EXTBLK         0000008C
CLI$_DEFKEY          = 0003DDC3                 PRC_L_EXTCOD         0000009C
CLI$_DELKEY          = 0003DDCB                 PRC_L_EXTHND         00000090
CLI$_IVKEYNAM        = 00038280                 PRC_L_EXTPRM         00000098
CLI$_KEYCNV          = 0003DE33                 PRC_L_IDFLNK         000000BC
CLI$_NORMAL          = 00030001                 PRC_L_IMGACTSTS      00000080
CLI$_SETKEY          = 0003DDD3                 PRC_L_INDCLOCK       0000007C
CLI$_SYMOVF          = 00038138                 PRC_L_INDEPTH        0000005C
CLI$_UNDKEY          = 00038260                 PRC_L_INDFAB         0000001C
COMMA                  00000071 R      02       PRC_L_INDINPRAB      00000014
DCL$ALLDYNMEM          ********    X   02       PRC_L_INDOUTRAB      00000018
DCL$ALLOCKEY           000005BD RG     02       PRC_L_INPRAB         00000008
DCL$ALLOC_STATE        00000794 RG     02       PRC_L_LASTKEY        0000004C
DCL$DEADYNMEM          ********    X   02       PRC_L_LSTSTATUS      000000B0
DCL$DEALLOCSYM         ********    X   02       PRC_L_ONCTLY         000000B8
DCL$DEALLOC_STATE      000007CA RG     02       PRC_L_ONERROR        0000006C
DCL$DEFKEY             000000C9 RG     02       PRC_L_OUTOFBAND      000000B4
DCL$DELKEY             00000205 RG     02       PRC_L_OUTRAB         0000000C
DCL$DISABLE            ********    X   02       PRC_L_OUTRABCTX      00000118
DCL$FIND_KEYPAD        000006DE RG     02       PRC_L_PPFLIST        00000070
DCL$FORMMSG            ********    X   02       PRC_L_RECALLPTR      0000012F
DCL$GETDVAL            ********    X   02       PRC_L_RESTART        00000058
DCL$GETNVAL            ********    X   02       PRC_L_SAVAP          00000000
DCL$LOCKED_STATE       ********    X   02       PRC_L_SAVFP          00000004
DCL$MSGOUT             ********    X   02       PRC_L_SEVERITY       00000050
DCL$SEARCH_KEYPAD      000006C1 RG     02       PRC_L_SPWN           000000C0
DCL$SETKEY             00000726 RG     02       PRC_L_STACKLM        000000A4
DCL$SHOWKEY            00000317 RG     02       PRC_L_STACKPT        000000A0
DCL$SYNONYM            00000651 RG     02       PRC_L_STATUS         00000054
DELKEY                 000002E2 R      02       PRC_L_STS            00000084
DISPHDR                00000583 R      02       PRC_L_STV            00000088
DISPSYMB               00000496 R      02       PRC_L_SYMBOL         00000060
E1_ADR                 000000B7 R      02       PRC_L_TMBX           00000074
E2_ADR                 000000BA R      02       PRC_L_TRMLIST        00000010
E3_ADR                 000000BD R      02       PRC_Q_ALLOCREG       00000020
E4_ADR                 000000C0 R      02       PRC_Q_COMMAND        000000E0
E5_ADR                 000000C3 R      02       PRC_Q_FLUSHTIME      000000D0
E6_ADR                 000000C6 R      02       PRC_Q_GLOBAL         00000028
FULLFAO                00000026 R      02       PRC_Q_IMAGENAME      000000D8
NO                     00000066 R      02       PRC_Q_KEYPAD         00000040
```

```
PRC_Q_LABEL                     00000030             WRK_B_MINPARM                   FFFFFFD1
PRC_Q_LOCAL                     00000038             WRK_B_PARMCNT                   FFFFFFCE
PRC_Q_SAVEPRIV                  000000E8             WRK_B_PARMSUM                   FFFFFFCF
PRC_T_OUTDVI                    0000011C             WRK_B_RECALLCNT                 FFFFFFC5
PRC_W_ASTIOSB                   000000C6             WRK_B_VALLEV                    FFFFFFC4
PRC_W_ASTRETN                   000000C8             WRK_B_VERBTYP                   FFFFFFC2
PRC_W_ASTSTATUS                 000000C4             WRK_C_INPBUFSIZ               = 00000100
PRC_W_ATTMBX                    0000007A             WRK_C_LENGTH                    FFFFF486
PRC_W_FLAGS                     00000068             WRK_G_BUFFER                    FFFFF492
PRC_W_INPCHAN                   00000064             WRK_G_INPBUF                    FFFFF896
PRC_W_ONLEVEL                   0000006A             WRK_G_RESULT                    FFFFF9B6
PRC_W_OUTIFI                    00000114             WRK_K_LENGTH                    FFFFF486
PRC_W_OUTISI                    00000116             WRK_L_CHARPTR                   FFFFF48E
PRC_W_OUTMBXCHN                 000000CA             WRK_L_DISALLOW                  FFFFFFE6
PRC_W_OUTMBXREF                 000000CE             WRK_L_ERRORRTN                  FFFFF9AE
PRC_W_OUTMBXSIZ                 000000CC             WRK_L_EXPANDPTR                 FFFFF486
PRC_W_PMPTCTRL                  000000F1             WRK_L_IMAGE                     FFFFFFE2
PRC_W_WAITIOSB                  00000066             WRK_L_MARKPTR                   FFFFF48A
PTR_B_LEVEL                     00000004             WRK_L_PAROUT                    FFFFFFD2
PTR_B_NUMBER                    00000005             WRK_L_PMPTADDR                  FFFFF9A2
PTR_B_PARMCNT                   00000006             WRK_L_PROMPTRTN                 FFFFF9A6
PTR_B_VALUE                     00000000             WRK_L_PROPTR                    FFFFFFC6
PTR_C_LENGTH                    0000000C             WRK_L_QUABLK                    FFFFFFCA
PTR_K_COMMA                   = 00000005             WRK_L_READRTN                   FFFFF9AA
PTR_K_ENDLINE                 = 00000004             WRK_L_RECALLPTR                 FFFFFFEA
PTR_K_LENGTH                    0000000C             WRK_L_RSLEND                    FFFFFFB6
PTR_K_PARAMETR                = 00000003             WRK_L_RSLNXT                    FFFFFFBA
PTR_L_DESCR                     00000000             WRK_L_SAVAP                     FFFFFFF8
PTR_L_ENTITY                    00000008             WRK_L_SAVFP                     FFFFFFFC
PTR_V_FLAGS                   = 00000014             WRK_L_SAVSP                     FFFFFFF4
PTR_V_KEYWORD                 = 00000015             WRK_L_SIGNALRTN                 FFFFFFD6
PTR_V_NEGATE                  = 00000014             WRK_L_SPECRTN                   FFFFF9B2
SHOWHDR                         00000000 R     02    WRK_L_TAB_VEC                   FFFFFFDE
STATE                           00000069 R     02    WRK_L_VERB                      FFFFFFBE
STS$M_INHIB_MSG               = 10000000             WRK_W_FLAGS                     FFFFFFF0
SYM_B_FLAGS                     0000000B             WRK_W_FLAGS2                    FFFFFFF2
SYM_B_NONUNIQUE                 0000000B             WRK_W_IMGCHAN                   FFFFFFEE
SYM_B_TYPE                      0000000A             WRK_W_PMPTLEN                   FFFFF99E
SYM_K_KEYPAD                  = 00000004             _$$_                          = 000000EF
SYM_L_BL                        00000004
SYM_L_FL                        00000000
SYM_M_ECHO                    = 00000001
SYM_T_SYMBOL                    0000000C
SYM_V_ECHO                    = 00000000
SYM_V_ERASE                   = 00000004
SYM_V_LOCK                    = 00000003
SYM_V_STATE                   = 00000002
SYM_V_TERMINATE               = 00000001
SYM_W_SIZE                      00000008
SYNDEF_TAB                      000000B7 R     02
SYNNAME_TAB                     00000073 R     02
SYS$FAO                         ********    X  02
SYS$FAOL                        ********   GX  02
UNDKEY                          000002FE R     02
VALIDATE_KEY_NAME               ********    X  02
WRK_B_CMDOPT                    FFFFFFC3
WRK_B_MAXPARM                   FFFFFFD0
```

```
                                                +----------------+
                                                ! Psect synopsis !
                                                +----------------+

PSECT name                      Allocation          PSECT No.  Attributes
----------                      ----------          ---------  ----------
.  ABS  .                       00000000  (     0.)  00 (   0.)  NOPIC    USR    CON    ABS    LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
$ABS$                           FFFFFFFC  (     0.)  01 (   1.)  NOPIC    USR    CON    ABS    LCL NOSHR   EXE   RD      WRT NOVEC BYTE
DCL$ZCODE                       000007E7  (  2023.)  02 (   2.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD    NOWRT NOVEC BYTE

                                             +------------------------+
                                             ! Performance indicators !
                                             +------------------------+

Phase                     Page faults    CPU Time        Elapsed Time
-----                     -----------    --------        ------------
Initialization                     10    00:00:00.07     00:00:00.47
Command processing                 87    00:00:00.72     00:00:03.56
Pass 1                            257    00:00:09.83     00:00:28.30
Symbol table sort                   0    00:00:00.87     00:00:02.66
Pass 2                            216    00:00:03.21     00:00:11.48
Symbol table output                24    00:00:00.17     00:00:00.66
Psect synopsis output               2    00:00:00.02     00:00:00.02
Cross-reference output              0    00:00:00.00     00:00:00.00
Assembler run totals              596    00:00:14.91     00:00:47.17

The working set limit was 1350 pages.
52334 bytes (103 pages) of virtual memory were used to buffer the intermediate code.
There were 40 pages of symbol table space allocated to hold 538 non-local and 89 local symbols.
1251 source lines were read in Pass 1, producing 20 object records in Pass 2.
44 pages of virtual memory were used to define 28 macros.

                                            +--------------------------+
                                            ! Macro library statistics !
                                            +--------------------------+

Macro library name                           Macros defined
------------------                           --------------
_$255$DUA28:[SYSLIB]SYSBLDMLB.MLB;1                      0
_$255$DUA28:[DCL.OBJ]DCL.MLB;1                          11
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                           0
_$255$DUA28:[SYSLIB]STARLET.MLB;2                        8
TOTALS (all libraries)                                  19

686 GETS were required to define 19 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:KEYPAD/OBJ=OBJ$:KEYPAD MSRC$:KEYPAD/UPDATE=(ENH$:KEYPAD)+EXECML$/LIB+LIB$:DCL/LIB+SYS$LIBRARY:SYSBLDMLB/LIB
```

0071 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

INQUIRE
LIS

LEXICON
LIS

KEYPAD
LIS

LOGICAL
LIS